ADOPTING AND PERFORMANCE EVALUATION OF WEBRTC-BASED CLOUD CPAAS: IN THE CASE OF ETHIOTELECOM

BY

BIRUK FIKADU GIZAW

OFFICE OF GRADUATE STUDIES

ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY

ADAMA, ETHIOPIA

JANUARY, 2021

Adopting and Performance Evaluation of WebRTC-based cloud CPaaS: In the case of Ethiotelecom

By

Biruk Fikadu Gizaw

Advisor

Dr. -Ing Frezewd Lemma

A Thesis Submitted to

Department of Computer Science and Engineering

School of Electrical Engineering and Computing

Presented in Partial

Fulfillment of the Requirements for the Degree of Masters of Science in

Computer Science and Engineering.

Office of Graduate Studies

Adama Science and Technology University(ASTU)

Adama, Ethiopia

January, 2021

Declaration

I, the undersigned, declare that the thesis comprises my own work in compliance with internationally accepted practices; I have fully acknowledged and referred all materials used in this thesis work.

Biruk Fikadu ____________________

Name Signature

Place: Adama

Date of Submission: ____________________

This thesis has been submitted for examination with my approval as a university advisor.

Frezewd Lemma (Dr.-Ing) ________________

Advisor Signature

Abstract

Cloud computing has become a leading technology solution for most of current problems in various sectors in terms of cost and security. Either in private enterprises or public domain cloud computing offers a vast amount of resource in computing, storage and network bandwidth. Presently most of service providers or telecoms such as broadband internet service providers suffer a lot from customer complaints using various cloud technology such as real time communication that are hosted outside the provider autonomous system. Due to various type of OTT (Over the Top) service over internet and other telecom frauds has a huge challenge in revenue, captical expenditure and operational expenditure. In this thesis research proposal, Adoption and Performance evaluation of web real-time communication (WebRTC) technology that's integrable with other serivices is presented as Communication platform as a service (CPaaS) in the case of Ethiotelecom. The proposed design evaluated in general QoS bandwidth and Routing complexities in enterprise network including a symmetrical NAT setup that will provide shortest path between peer-to-peer communications. CPaaS cloud provisioning algorithm presented using Cloudsim plus simulator for various types of virtual datacenters and resource. The paper tries to depict that cloud communication platform as a service can be beneficial for internet service provider like Ethiotelecom without compromising their core service performances. The test results have displayed reduced NAT (Network Address Translation), low bandwidth consumption on the provider network, reduced congestion on internet gateways with the implementation of WebRTC inside ISP. Finally, WebRTC implementation from grean computing perspective, allows ISP use their resource efficiently by incorporating informaction technology by still maintaining the overall service performance.

Keywords: CPaaS, Cloud, WebRTC, VoIP, OTT, NAT

Acknowledgments

I would first like to thank my thesis advisor Dr. Ing Frezewd Lemma of the School of Electrical Engineering and Computing at Adama Science and Technology University. The door to Dr. Ing Frezewd office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work, but steered me in the right the direction whenever he thought I needed it.

I would also like to thank the colleagues and experts at Ethiotelecom who were involved in the validation inputs for this research project. Without their passionate participation and input, the research could not have been successfully conducted.

I would also like to acknowledge Dr. Mesfin Abebe of the School of Electrical Engineering and Computing at Adama Science and Technology University in his supporting, and I am gratefully owing gratitude for his very valuable supports on the thesis.

Finally, I must express my very profound gratitude to almighty God and also my lovely wife and dearest mother for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Biruk Fikadu

List of Abbreviations and Acronyms

API	Application Program Interface
AS	Autonomous System
BGP	Border Gateway Protocol
BR	Backbone Router
CAPEX	Capital Expenditure
CDN	Content Delivery Network
CPaaS	Communication Platform as a Service
CR	Core Router
CS	Circuit Switched
DOM	Document Object Model
DNS	Domain Name System
DSRM	Design Science Research Methodology
ETSI	European Telecommunications Standards Institute
GPRS	General Packet Radio Service
GPS	Geographical Position System
GW	Gate Way
HD	High-definition
HSDPA	High-Speed Downlink Packet Access
HSPA	High-Speed Packet Access
HTML5	Hypertext Markup Language 5
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICE	Interactive Connectivity Establishment
ICT	Information Communication Technology
IETF	Internet Engineering Task Force
IMS	IP Multimedia subsystem
IP	Internet Protocol
ITU	International Telecommunication Union
ITU-T	ITU Telecommunication Standardization Sector
Kbps	Kilobits Per Second
KPI	Key Performance Indicators

KQI	Key Quality Indicators
Mbps	Mega Bits per second
MGW	Media Gate Way
Ms	Millisecond
MSC	Mobile Switching Center
NAT	Network Address Translation
NGN	Next Generation Network
NMS	Network Management System
NNOC	National Network Operation Center
OTT	Over the Top
OS	Operating System
PAT	Port Address Translation
PCs	Personal Computers
PS	Packet Switched
PSTN	Public Switched Telephone Network
QoE	Quality of Experience
QoS	Quality of Service
RFC	Request for comment
RAN	Radio Access Network
RTP	Real-Time Protocol
RTSP	Real Time Streaming Protocol
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SMC	Service Management Center
STUN	Session Traversal Utilities for NAT
TCP	Transmission Control Protocol
TURN	Traversal Using Relays around NAT
UDP	User Datagram Protocol
URL	Universal Resource Locator
VoD	Video-on-Demand
VoIP	Voice over IP
VM	Vertual Machine
W3C	World Wide Web Consortium

Wi-Fi	Wireless Fidelity
WebRTC	Web Real-Time Communication

Table of Contents

Abstract i
Acknowledgments ii
List of Abbreviations and Acronyms iii
Table of Contents vi
List of Figures x
List of Tables xii
CHAPTER 1 1
Introduction 1
1.1 Background 2
1.2 Motivation 2
1.3 Statement of Problem 3
1.4 Research Questions(RQs) 5
1.5 Objective 6
1.5.1 General Objective 6
1.5.2 Specific Objective 6
1.6 Scope and limitation of study 6
1.6.1 Scope of the Study 6
1.6.2 Limitations of the Study 6
1.7 Application of Results 7
1.8 Methodology 7
1.9 Thesis Organization 8
CHAPTER 2 9
Review of Literature and Related Works 9
2.1 Web Real Time Communication 9
2.1.1 Multi-user Video WebRTC Streaming 9
2.2 WebRTC Signaling 12

2.2.1 WebSocket 12
2.2.2 Session Initiation Protocol (SIP) 12
2.2.3 Session Description Protocol (SDP) 12
2.2.4 ICE, TURN/STUN 13
2.3 NAT (Network Address Translation) 14
2.4 Web RTC Media 15
2.4.1 Real Time Protocol (RTP) 15
2.4.2 Real Time Control Protocol (RTCP) 15
2.4.3 Over the top (OTT) 16
2.4.3.1 Telecom Fraud and Over-The-Top service 16
2.5 Internet Service Provider network 18
2.5.1 Cloud Communications platform as a service 20
2.5.2 Public vs Private vs Hybrid cloud services 20
2.5.3 Internet Network VoIP QoS analysis 22
2.5.4 Related Work 26
CHAPTER 3 30
Proposed WebRTC CPaaS Model 30
3.1 WebRTC CPaaS Architecture overview 31
3.1.1 Cloud Infrastructure for PaaS 32
3.1.2 Cloud Platform Runtime for CPaaS 32
3.1.3 Cloud DCN and Consumer network 33
3.2 WebRTC Signaling and Gateway 33
3.2.1 ICE STUN/TURN Server 34
3.3 Algorithm 37
3.3.1 New CPaaS provisioning resource allocation 37
3.3.2 Scaling CPaaS provisioning resource 39
3.3.3 Terminiation of cloud resource 41

3.4 Cloudsim Plus Cloud simulator 43
CHAPTER 4 45
Experiment Setup and Evaluation 45
4.1 Overview 45
4.2 Hardware 45
4.3 Software 45
4.3.1 WebRTC App User interface video GUI 46
4.3.2 WebRTC API code 48
4.4 Experiment setup 49
4.4.1 Dummynet configuration and WebRTC analyzer 49
4.4.2 Case I:WebRTC connection with external STUN 50
4.4.3 Case II:WebRTC connection setup with local STUN 51
4.4.4 Case III:WebRTC connection relay with local TURN server 53
4.4.5 Case IV:WebRTC connection relay with external TURN 57
CHAPTER 5 59
Results and discussion 59
5.1 Overview 59
5.2 WebRTC with external STUN server 59
5.3 WebRTC with local STUN server 60
5.4 WebRTC connection relay with local TURN 60
5.5 WebRTC connection relay by external TURN 61
CHAPTER 6 62
Conclusion and Future Work 62
6.1 Conclusion 62
6.2 Contributions 63
6.3 Recommendations for Future work 63
References 64

APPENDIX 69
6.4 Appendix 69
6.4.1 Appendix A: WebRTC Signaling STUN & TURN setup 69
6.4.2 Appendix B: CloudSim Plus new Data Center Provisioning example code and Output 70

List of Figures

Figure 1.5.1-1 SIP code embedded inside web page javascript code[43] 4
Figure 1.6.2-1 Design Science Research Methodology Process Model. [7] 8
Figure 1.6.2-1 WebRTC supported by many browsers and OS's.. 9
Figure 2.1.1-1 WebRTC Mesh Network.. 10
Figure 2.1.1-2 Selective Forwarding Unit connection... 11
Figure 2.1.1-3 MCU connection for multi-users 12
Figure 2.2.4-1 Stun request and response scenario.. 14
Figure 2.4.1-1 RTP header.. 15
Figure 2.4.2-1 WebRTC Protocol stack... 16
Figure 2.4.3-1 OTT bypass fraud scenario (Image source www.commsrisk.com) 17
Figure 2.4.3-1 Internet and Various ISP connectivites(source [21]) 19
Figure 2.5.1-1 IaaS, PaaS & SaaS Cloud types management by provider and user................ 20
Figure 2.5.2-1 Communication platform as a service... 21
Figure 2.5.3-1 Users connecting to various OTT service providers....................................... 25
Figure 3.1.1-1 CPaaS cloud for a type-1 hypervisor layers.. 32
Figure 3.1.3-1 WebRTC Signaling and Gateway.. 34
Figure 3.2.1-1 ICE communication flow using STUN.. 35
Figure 3.2.1-2 TURN server communication... 36
Figure 3.3.1-1 New CPaaS request process flow.. 39
Figure 3.3.2-1 CPaaS Scaling upgrade downgrade request handler flow chart....................... 41
Figure 3.3.3-1 CPaaS Termination Request Handler flow chart 43
Figure 3.3.3-1 CloudSim simulation engine. 44
Figure 4.3.1-1 Demo application structure 47
Figure 4.3.1-2 Login registration page 47
Figure 4.3.1-3 WebRTC application main page 48
Figure 4.4.2-1 WebRTC App test using external stun server.. 51
Figure 4.4.3-1 Experiment 2 Topology setup 52
Figure 4.4.3-2 WebRTC Video Bitrate call between user1 and user2 using local STUN ICE52
Figure 4.4.3-3 RTC outbound RTP Audio Bitrate.. 53
Figure 4.4.3-4 Video jitter for incoming and outgoing traffic with local STUN server.......... 53
Figure 4.4.3-5 Video Average Round Trip Time for Local STUN server 53
Figure 4.4.4-1 Firefox browser WebRTC relay only configuration. 54

Figure 4.4.4-2 Forced TURN server Relay topology using local TURN server 55
Figure 4.4.4-3 WebRTC call Video Bitrate using forced local TURN ICE 55
Figure 4.4.4-4 WebRTC incoming and outgoing Audio Bitrate for local TURN server 56
Figure 4.4.4-5 WebRTC jitter for incoming and outgoing traffic using local TURN server . 56
Figure 4.4.4-6 WebRTC average round trip time using local TURN server........................... 56
Figure 4.4.5-1 Forced TURN server Relay topology using external TURN server 57
Figure 4.4.5-2 WebRTC Video Bitrate using external TURN ICE... 58
Figure 4.4.5-3WebRTC Packet loss using forced external TURN ICE 58
Figure 4.4.5-4 WebRTC Video Jitter using forced external TURN ICE................................. 58
Figure 4.4.5-5 WebRTC Video Average RTT using external TURN ICE............................. 58

List of Tables

Table 2.5.3-1 Delay spacifications for applications 23
Table 2.5.3-2 Delay for interactive applications 24
Table 2.5.4-1 Related works 29
Table 4.4.2-1 External STUN server configuration 50
Table 4.4.3-1 Local ICE server configuration 51
Table 4.4.4-1 Local TURN server configuration 54
Table 4.4.5-1 Output between two devices communication via STUN server. 60
Table 4.4.5-1 Avarage output for a peer-to-peer communication via TURN server. 61

CHAPTER 1

Introduction

WebRTC is a plugin free and supported by most of present web browsers such as Apple, Google, Microsoft, Mozilla, and Opera, WebRTC is being standardized through the W3C (World Wide Web Consortium) in cooperation with RTCWeb protocols developed by the Internet Engineering Task Force (IETF). WebRTC is also OS independent runs on most operating systems such as mac, android, Linux and windows which make is preferable than other multimedia applications.[1]

Internet and data communications have evolved a lot of major changes and currently a global real time communication is possible instantly anywhere in the world. Internet from its beginning not directly developed for a real time communication especially for packet switched network. Internet communication which is usually a packet switched network that uses mostly IPv4 addressing scheme have most of hosts or devices behind different types of NAT(Network Address translation) router. [2] For a real-time communication setup there are a lot of set of protocols, paths and connection media types in which a datagram or packet travels from sender to the receiver in a fraction of time, milliseconds usually.

Video and Voice over IP services are the choice of many communication users and business enterprises, they prefer such as. Skype, WhatsApp, Google Hangouts, Viber and etc. Which are an internet communication services requiring a high bandwidth that are highly affecting the legacy or old telecom service provider that owns a network infrastructure in quality of service (QoS) and revenue. ISP usually suffers from degraded QoS complaint from broadband customer that are using real-time communication that are provided outside the service provider domain. A real-time communication over internet is sensitive by nature and requires end-to-end resource allocation or conditions that are suitable for synchronous communication flow.

Developing a cloud solution by individual enterprises alone is highly costly investment that requires infrastructure, skill and operational cost. And also, the existing internet multimedia subsystem (IMS) by the most service provider doesn’t fulfil the demand of users most of the time. There are a variety of VoIP applications and systems such as cisco VoIP, opensource VoIP applications with a limited feature and most of the time with compatibility problem.

1.1 Background

ISP's or telecom operators provide a wide verity of communication and multimedia services to fulfil their individual and enterprise customers' needs or demands. Besides the legacy fixed network telephone line, mobile and internet connection service for communication, many enterprise and entrepreneurs are demanding and using cloud solutions. Most traditional telecommunication service providers, such as Ethiotelecom are trying to catchup with recent technologies and innovations.

ISP-OTT service providers currently offers a variety of services such as voice or video call, file transfer, text etc. services at a cheap price without any regulation or security authentication. OTT providers are usually based in different locations or country that doesn't contribute in revenue or tax to the infrastructure owned by traditional telecom service provider. These trends have gradually have led to the situation in which the data and voice tariffs are incomparable in traditional telecoms. Therefore, users prefer the cheap data service in order to communicate via internet video audio over IP service provided by on the top (OTT) service providers that resides outside their service providers autonomous system.[3]

1.2 Motivation

Cloud and distributed system computing paradigm are widely implemented throughout the world, but Ethiopia's ISP lags behind. Governmental, non-governmental organization and private enterprises such as education, healthcare's, manufacturing industries, and etc. have their own data centers, network infrastructures which are not distributed and not utilized efficiently.

In case of Ethio telecom as an example, I have got the exposure in Ethio telecoms' current data centers as a professional employee with NGN (Next Generation Network). Ethio telecom data center project and other network projects and operational activities. Ethiotelecom as a service provider have a huge infrastructure network that can cover more than 85% of the national coverage demands, but the company lags in implementation of various types cloud and information communication technologies (ICT) solutions to its customers.

Hence, this thesis is motivated by the following five key ideas:

- Cloud computing future tendency of largely implementation for business in the country.
- Wide demand of cloud ICT applications, OTT, Ads service by individual customers and business enterprise requirement.

- Real time services which are domestically(locally) generated packet data traffics that are destined to local but uses external OTT servers must have alternatives and QoS.
- Opportunities needed to be provided for developers, multiple tenants and ISP-OTT service providers with business and revenue sharing.
- Cloud computing provide a huge benefit for development of nation in education, agriculture, technology, industrial manufacturing etc. among all communication platforms plays key role.

1.3 Statement of Problem

Despite Addis Ababa, being the world's top diplomatic capital; there is no service offering of online real time video, audio conference streaming service, and video/audio IMS (IP multimedia subsystem) from the sole telecom service provider or any other based in Ethiopia.

Even if Ethiopia is able to cover most of its sectors, government and public service with intra-network and internet; however, it is yet to harvest the benefits of ICT technology. Most of civil service and ICT professionals are poor in skills for ICT usage and IT knowledge which needs a lot of up-to-date training, seminars and education in which needs a large amount of cost for training and development program.

The demand of real time communication video/audio and other IMS services have increased more and more through time in Ethiopia, such as on-job training, conferences, distance education, web-chatting, remote-business support and other sensitive services requires real time communication. In comparison to other countries telecom network Ethiopian telecom price are not said to be cheap [4] in internet price. Also, as a service provider the company lags behind in cloud computing, web hosting and other general IMS service. Problems associated with services that are provided from OTT providers or outside the telecom (ISP) without any alternatives usually resulted in customer dissatisfaction and huge compliant [5]

Current situation in Ethiotelecom networks

Constantly increasing volumes of traffic due to the needs for mobile data services and Internet use continues to increased rapidly that affected the quality of service and bandwidth congestion at internet gateway routers. Also, the growth of new types of traffic such as video, online games, voice, real time and online interactive communication and applications that cannot be cached in order to reduce outward traffics in which makes it difficult for real-time communication such as video call.

Servers located outside the operator's network (can be outside continent, Transoceanic or Satellite Links): Many companies set up multiple servers in data centers around the world and use a load balancer to route traffic to the nearest or less busy server. Besides the latency and jitters, it's more possible to be tracked by third party sites or users not being comfortable with their data being stored in other countries.

Most Internet service providers problem is IPv4(IP address version 4) being used almost fully and a few address remaining unallocated and the expected IPv6(IP version 6) migration not fully implemeted. These and other problems has lead to another solution which is NAT(Network address Translation) in which one public IP can cover large number internal network in other term call local area network. Transfer of several types of traffic in one session in order to save ipv4 address, there are various problems associated with multiple NAT and PAT(Port Address Translation) problem whenever the traffic size increases some traffics are interrupted by policies at firewalls.

Constantly growing range of added services and user applications, with the business opportunities missed that provides a revenue and satisfaction for the connected customers with a minimal CAPEX (Capital expenditure) and OPEX (Operational expenditure). The main challenges also have been using the network infrastructure capacity for domestic network demands and ICT solutions that doesn't require internet connection. So, the research for alternative ICT solution and a wide variety of communication platforms choices are essential.

In a recent year, introduction of WebRTC (Web real-time communication) is becoming a more uncomfortable reality for traditional operators due to the fact that any peer to peer ipv4 communication can happen with appropriate signaling across the world. Various mobile applications requiring privacy permissions including managing phone calls in which any developer can put simple code into their web page which look like this;

```
$.phono ({
onReady: function (){
  this.phone.dial("sip:9999123135@example.net")
}});
```

Figure 1.5.1-1 SIP code embedded inside web page javascript code[43]

Since Web real time communication is getting a huge momentum used by a huge clouds and social networks such as google hangouts and meet, facebook messanger, discored, amazon chime etc. web real time communications audio as well as video become a huge demand for internet users due to cheep price in data than voice service from their internet providors. Based on this conditions telecom providers such as Ethiotelecom face a huge challagens from broadband as well as narrow band users complaining about OTT provided real time communications by associating the internet speed quality.

The main challage for WebRTC is that its open signaling complexity and usually offered by OTT providers in which they doesn't have a dedicated network infrastructure and use a software based packet forwarding. Today many telecom providers still doesn't have their free or commercial Interactive Connectivity Establishment protocol(ICE) STUN and TRUN signaling services in which OTT providers use external such as google stun server.

1.4 Research Questions(RQs)

The thesis generally covers the answers of the following research questions:

RQ1: What are the main issues related to Web Real Time Communication in terms of ISP or telecom service providers and trends in Ethiotelecom?

The paper covers the challenges and opportunities related to WebRTC on telecom (ISP's) services providers and OTT service providers. Real time communication over the internet in a packet switched network requires security and as well as quality at the same time. Voice and Video services usually provided by OTT service providers have highly affected the legacy communication trends with share revenue covered in section 2.4.3 under OTT.

RQ2:What problem lies on existing Real-time communications over the Internet as well as ISP?

Services provided by OTT providers and some social medias are free or cost lesss amount as compared to legacy ISP' but they don't have control over network infrastructure or their traffics relies on best effert of software defined network routing.

RQ3:How to evaluet performance of WebRTC in terms of various network scenarios and parameters?

The main focus of this thesis paper is to adapting WebRTC technology as a cloud communication service such that the ISP benefits from high quality, less traffic on gateways and backbone routes and business opportunites. The numberical parameters used in the reseach are bandwidths, jitter, audio/video bit rate packet loss and video resolution. Using different network topology scenario results are compared using practical and analytical tool to evaluet performance of WebRTC. The result and performance anylysis presented in chapter 5 in detail.

1.5 Objective

1.5.1 General Objective

The general objective of this research is Adoption and Performance Evaluation of WebRTC-based cloud CPaaS for VoIP and OTT Challenges in the case of Ethiotelecom, in such a way that reduces traffic and revenue problems.

1.5.2 Specific Objective

The specific objective of this research is as depicted as follows:

- Design a scalable cloud based WebRTC CPaaS environment
- Provide conceptual framework for implementation of the cloud based WebRTC
- Provide a minimal routing overhead and minimum bandwidth consumption on ISP backbone network for local traffics.
- Easy troubleshooting maintenance and customization of WebRTC
- A P2P Simulation of WebRTC Perfomance Evalution
- Provide result and analysis from test

1.6 Scope and limitation of study

1.6.1 Scope of the Study

The Scope of this thesis is bound to only peer-to-peer video and voice part of WebRTC application in general due to the depth of WebRTC technology collection of protocols. WebRTC distance and Internet factors that affects a peer to peer video audio call will be analyzed. Since real time communication over internet is highly dependant on the quality of internet links such as bandwidth and latency special emphasis is given to it in WebRTC.

Since WebRTC technology requires an input media devices for video and audio calls makes it difficult for simulation on a virtual environment thus, actual testing machines will be used such as two Laptops or mobile devices.

1.6.2 Limitations of the Study

Due to the requirement of a large computing resource requirements for cloud stacks and cloud hypervisours the study limted to testing the CPaaS application on guest OS using a type 2 hypervisor. Also the research experiment focused only on a peer to peer video call between to user ends.

1.7 Application of Results

The primary result of this research is beneficial to Ethiotelecom as a service provider in pointing solutions to the problems faced and chance for new business as a best telecom service provider.

Therefore, in the first place as a developing country Ethiopia needs and will be beneficial from WebRTC and second as its revenue and value-added service Ethiotelecom will be very much beneficial in applying result of this research. However; in consideration of with the existing network infrastructures and connectivity in Ethiopia and the capability of Ethiotelecom as a service provider. Considering Health-net, Wereda-Net, School-Net, University-Ether-Net and Other connections it's already place to unleash the power of real time communication.

WebRTC as a cloud CPaaS can be implemented for:

WebRTC Education: - Distance education, seminars, conference and short/on job training can be offered with high quality voice and audio.

WebRTC Call Center: - WebRTC can be used as an online real-time interaction between customers and a call center support for various types of marketing, service and online business.

WebRTC e-visit or online health: WebRTC can be usefull for online healthcare service in which doesn't require a physical examination.

WebRTC for security and smart home: WebRTC can be used for remote management and security also can integrated with other device such as android.

WebRTC for the Internet of Things: for future implementation of various IO devices.

1.8 Methodology

To conduct this thesis research in accordance to the aim and objective of the research, two types of research methods are used. First, the WebRTC OTT VoIP is evaluated in terms of connection time and quality of service using analytical method.

Then by applying Design Science Research Methodology with additional evaluative steps WebRTC CPaaS is presented with a minimum packet on ISP network and minimum latency. In DSRM the initial step is to identify the problems and motivation followed by defining the objective of the solution. Design science research (DSR) methodology stress on the development and functioning of design artifacts with the detail aim of exceeding the functional performance of the artifact. Design science research is mostly applied to a class of artifacts including human/computer interfaces, algorithms, design methodologies (including process models) and languages. [6]

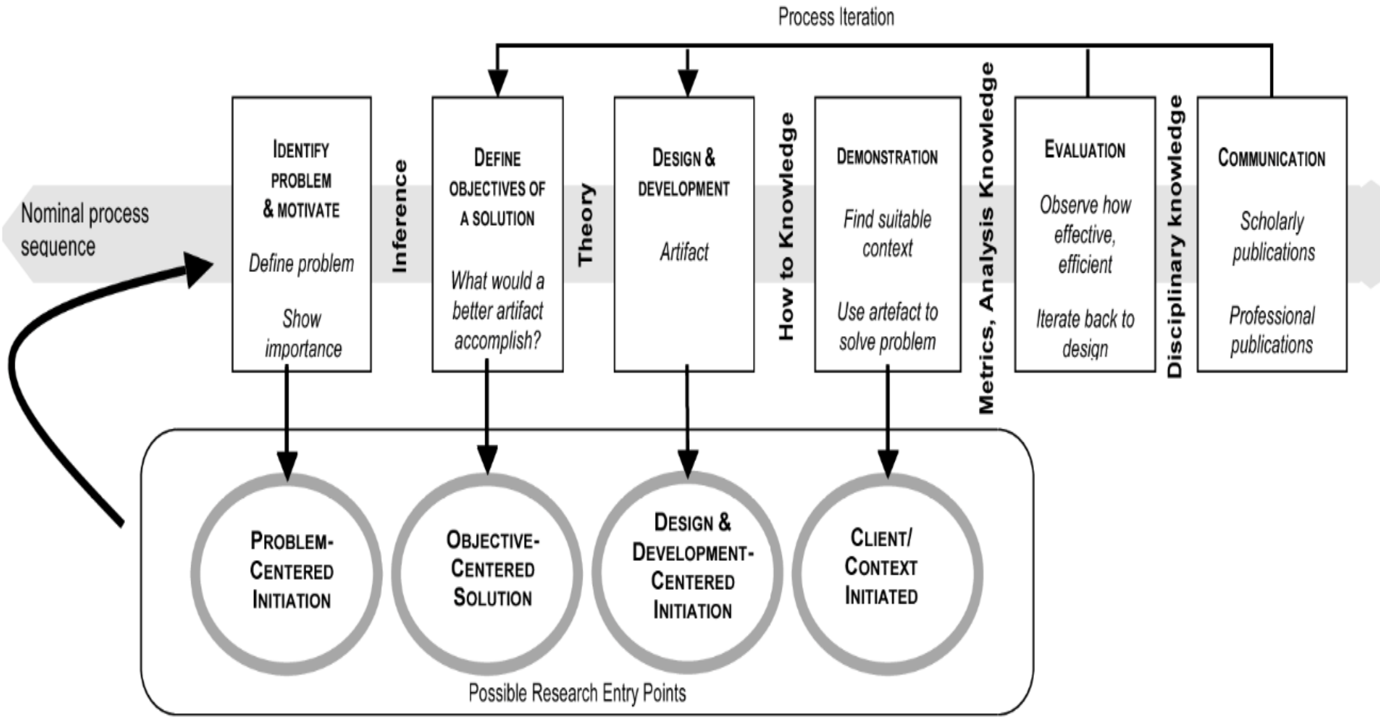

Figure 1.6.2-1 Design Science Research Methodology Process Model. [7]

1.9 Thesis Organization

This thesis is organized as follows:

Chapter 2: Discuss in detail the Background of the RTC and WebRTC and also explains various WebRTC architectures and WebRTC components and explains the related works that are carried out in the WebRTC and RTC

Chapter 3: Presents WebRTC CPaaS proposed solution

Chapter 4: Details the implementation experiments carried in this thesis and presents the results of the Experiments.

Chapter 5: Provides Result and Discussion of the Test case scenarios

Chapter 6: Conclusion and Future work will be presented

CHAPTER 2

Review of Literature and Related Works

2.1 Web Real Time Communication

For a long time, Web browser are used mostly in a one directional way a client-server architecture with web server but with HTML5 and WebSocket protocol web browser now has the ability of bi-directional communication. WebRTC is a set of APIs and Protocols that enables two web browsers the real-time communication feature such as audio, video and instant massaging using WebSocket protocol.

Figure 1.6.2-1 WebRTC supported by many browsers and OS's

2.1.1 Multi-user Video WebRTC Streaming

Basically webrtc only works in a peer to peer connection by default it doesn't have native support for multi-user video chat streaming. Thus, there are three mechanisms in which a WebRTC supports multi-user video chat is for few users Mesh connection, Selective Forward Unit(SFU) and Multipoint control Unit(MCU). [10]

Full Mesh Connection

In full mesh connection WebRTC implements a distinctive peer to pear connection between each users so that video stream will be sent for each individual peer connected and received from each. In a full mesh network with n number of peers their will be n(n-1) links however; its disadvantage is as the number of participating peers increase CPU utilization will spike and network might get congested, but it can achieve low latency for few peers.

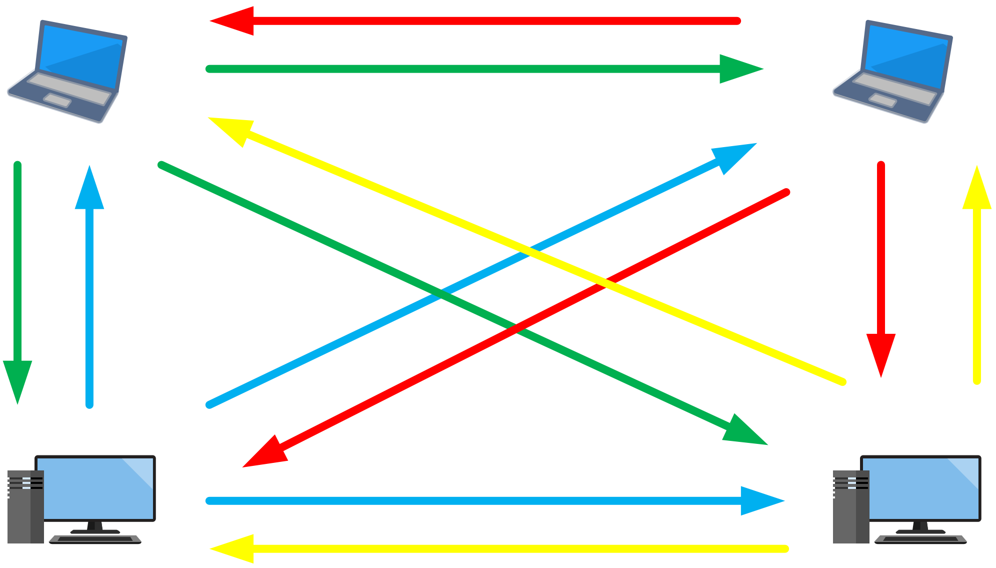

Figure 2.1.1-1 WebRTC Mesh Network

Selective Forwarding Unit

In Selective Forwarding case each peer will send their video stream to the SFU server and server in turn will send it to others via the same server. The SFU architecture enables the participant peers to send multiple video streams to the SFU, then the SFU will select how the video streams would be forwarded to any participants peer.

SFU does not involve in decode and re-encode received video streams unlike in the MCU, SFU simply acts as a forwarder of streams between participant peers. SFU works with asymmetric bandwith and have scalability which the two best attributes of SFU.   [11]

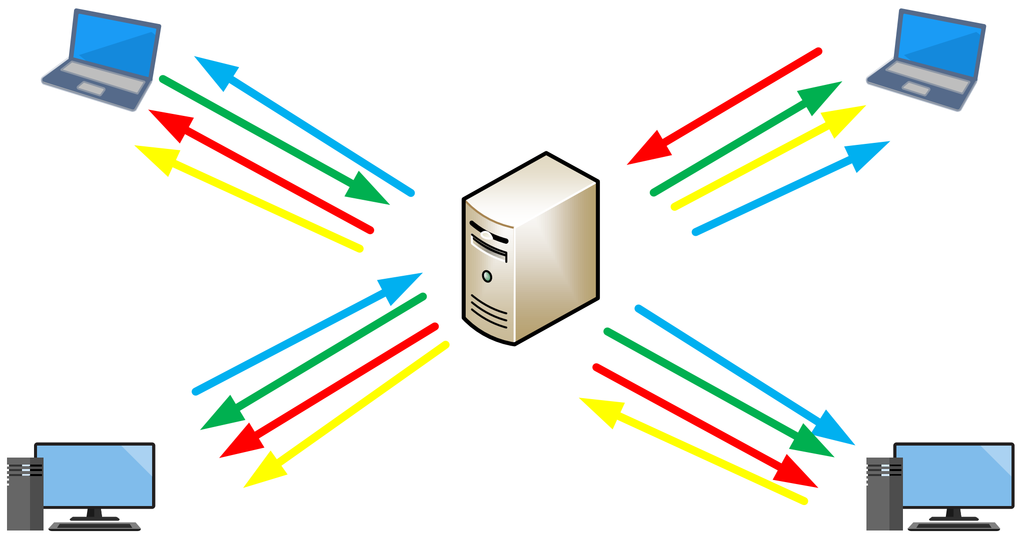

Figure 2.1.1-2 Selective Forwarding Unit connection

Multipoint Control Unit

MCU device works by decoding video or media streams received into a merged single video by rescaling it or process it so each end user will view merged video and independent audio. MCU device is a video conferencing device that connected a VoIP terminals connected by participation into a multi-party video conference system. It will be far more effective on bandwidth usage (each peer only upload once, and download once), and indeed there is a few more utilization on CPU.

MCU allows the participant to control different options. Types include:

- Individual user can be displayed on a different screen.
- Voice-activated selection of displayed speakers.
- Screenshare

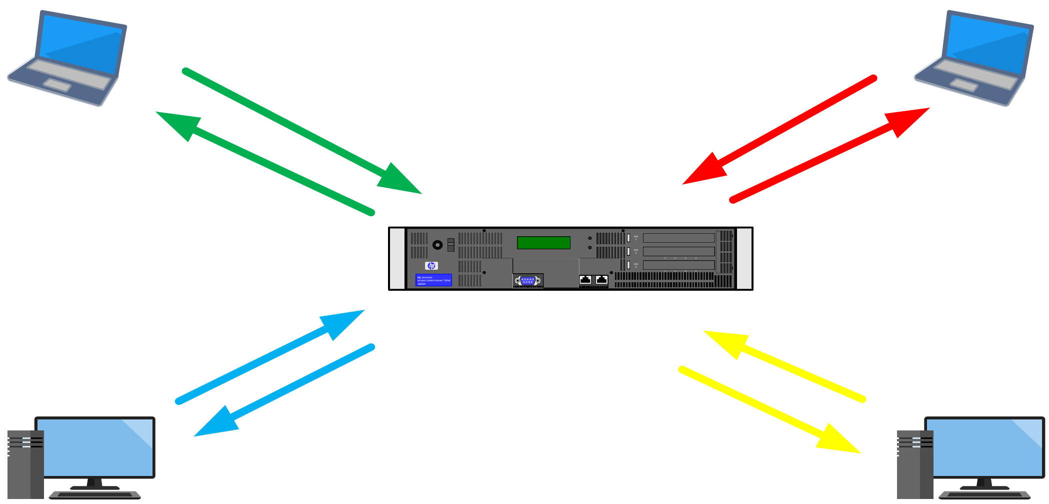

Figure 2.1.1-3 MCU connection for multi-users

2.2 WebRTC Signaling

WebSocket protocols in web browser enables the bi-directional transmission possible in a peer-to-peer or client server communication in real-time. Thus, using the WebSocket protocol two clients in the same local area network (LAN) can communicate with less complexities i.e. sending Session initiation protocol SIP, session description protocols SDP about their parameters such as video audio codec types, however if they are on different network WebRTC requires route assistant Interactive Connectivity Establishment (ICE) RFC-8445 protocol. [12]

2.2.1 WebSocket

A WebSocket is a communication protocol providing a full-duplex persistent connection between a client and server. WebSocket protocol is usually used to send signals from server to client for a real-time updates. WebSocket API is widely used by many real time applications such as gaming, since it makes possible to open a two-way interactive communication between the user's web browser and a server. Websock in WebRTC allows a biderctional massage transmission between two browser in the internet to communicate their ICE candidate and SDP message the will communicate such as media codic. IETF standardized WebSocket protocol RFC 6455 in 2011, and W3C standardized.[13]

2.2.2 Session Initiation Protocol (SIP)

Session Initiation Protocol (SIP) is a signaling protocol to initiate an IP communication session between two user devices on a network. SIP is most popular and widely used signaling protocol in VoIP and other communication technologies. SIP is a text based protocol which makes it easy to capture each SIP packet and understand and troubleshoot.[14]

JSEP (java script session establishment protocol) is a IETF draft standard for WebRTC for separting signaling and media messages into different layers. JSEP come into play when signalling through a W3C RTCPeerConnectionAPI in starting a multimedia session. JSEP also hadles the communication between ICE stun and turn servers in peer discovery. [15]

2.2.3 Session Description Protocol (SDP)

The Session Description Protocol (SDP) is a general protocol that can be used in a different types of situations and is typically used in conjunction with one or more other protocols (e.g., SIP). It conveys the following information:

- The type and purpose of the session
- Start and end times for the session
- The media types (e.g., audio, video) that comprise the session
- Detailed information required to receive the session (e.g., the multicast address to which data will be sent, the transport protocol to be used, the port numbers, the encoding scheme)

SDP provides this information formatted in ASCII using a sequence of lines of text, each of the form “<type> = <value>.” An example of an SDP message will illustrate the main points.

```
v=0
o= addis 2890844526 2890842807 IN IP4 196.189.197.1
s=Networking 101
i=C class on computer networking
u=http://webrtc.test/
e=addis@astu.edu.et
c=IN IP4 197.189.197.1
t=2243252356235622
m=audio 59870 RTP/AVP 0
m=video 61372 RTP/AVP 31
m=application 842416 udp wb
```

2.2.4 ICE, TURN/STUN

Interactive Connectivity Establishment (ICE) is a blanket standard that describes how to coordinate STUN and TURN to make a connection between hosts. Network Traversal Service implements STUN and TURN for ICE-compatible clients, such as browsers supporting the WebRTC standard.

A host uses Session Traversal Utilities for NAT (STUN) to discover its public IP address when it is located behind a NAT/Firewall. When this host wants to receive an incoming connection from another party, it provides this public IP address as a possible location where it can receive a connection. If the NAT/Firewall still won't allow the two hosts to connect directly, they make a connection to a server implementing Traversal Using Relay around NAT (TURN), which will relay media between the two parties. Example stun request response can be shown as,

```
User                          NAT                       STUN server
      |------ STUN request ----->|                          |
      |   src:192.168.1.2:3000   |                          |
      |                          |------ STUN request ----->|
      |                          |   src:213.55.189.9:8888  |
      |                          |                          |
      |                          |<------STUN response------|
      |                          |   ext:213.55.189.9:8888  |
      |<------STUN response------|                          |
      |   ext:213.55.189.9:8888  |                          |
```

Figure 2.2.4-1 Stun request and response scenario

STUN, TURN, and ICE are a set of IETF standard protocols for negotiating traversing NATs when establishing peer-to-peer communication sessions. WebRTC and other VoIP stacks implement support for ICE to improve the reliability of IP communications.

2.3 NAT (Network Address Translation)

Firewalls introduce a single check in point in a network where a security policy can be enforced, this logically divide another space an internal network trusted network and an external untrusted network. There is another method we can make use of the idea of internal and external network and it’s typically implemented in the network access router network address translator(NAT). NAT is a way to map and IP address from a certain sub network from internal network to an external network. The main goal of NAT now a day is solving the depletion of IP version 4 address. NAT indirectly provides security where devices in the internal network are not directly accessible by the external network. NAT works by accessing and manipulating network and transport level packet information. NAT implements several types of restriction

Full cone NAT: know as static NAT or one-to-one NAT if a connection form a private ip any port is open towards open internet NAT creates a mapping between internal private ip port and public ip port and every subsequent packet from the same source private ip port will be forwarded to open internet as a public ip port pair. Full cone NAT allows every connection with a destination public ip port pair form any source port in the internet to be forwarded to private ip port, this could mean replie traffic form the just opened session or any other in coming traffic. Full cone NAT is the most permissive NAT type.

Restricted cone NAT: works the same way as a full cone NAT, but implements additional mechanisms on which external ip can communicate with the internal private ip address.

Port Restricted cone NAT: As restricted cone NAT checks the source ip from the public ip address port restricted NAT implements port checking mechanisms.

Symmetric NAT: Allows all requests from internal private IP address and port, to a specific destination internet IP address and port , are mapped to the same external public IP address and port.[16]

2.4 **Web RTC Media**

WebRTC uses a real time protocol (RTP) for delivering video audio traffic over IP network. RTP typically runs over User Datagram Protocol (UDP). RTP is used in conjunction with the RTP Control Protocol (RTCP). While RTP carries the media streams (e.g., audio and video), RTCP is used to monitor 6transmission statistics and quality of service (Qu's) and aids synchronization of multiple streams. RTP is one of the technical foundations of Voice over IP and in this context is often used in conjunction with a signaling protocol such as the Session Initiation Protocol (SIP) which establishes connections across the network.

2.4.1 **Real Time Protocol (RTP)**

The Real-time Transport Protocol (RTP) provides a framework for delivery of audio and video teleconferencing data and other real-time media applications. Previous work has defined the RTP protocol, along with numerous profiles, payload formats, and other extensions. When combined with appropriate signaling, these form the basis for many teleconferencing systems. [17]

The RTP header has the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.4.1-1 RTP header

The first twelve octets are present in every RTP packet, while the list of CSRC identifiers is present only when inserted by a mixer. [18]

2.4.2 **Real Time Control Protocol (RTCP)**

Real-time Transport Control Protocol and is defined in RFC 3550. RTCP works hand in hand with RTP. RTP does the delivery of the actual data, whereas RTCP is used to send control packets to participants in a call. The primary function is to provide feedback on the quality of service being provided by RTP.

RTP is originated and received on even port numbers, and the associated RTCP communication uses the next higher odd port number. It transports statistics and information such as octet and packet counts, jitter, and round-trip time. An application can use this information to control QoS parameters and choose, for example, to use a different codec. RTCP does not provide any flow encryption or authentication methods, but such mechanisms can be implemented with the use of the Secure Real-time Transport Protocol (SRTP). [19]

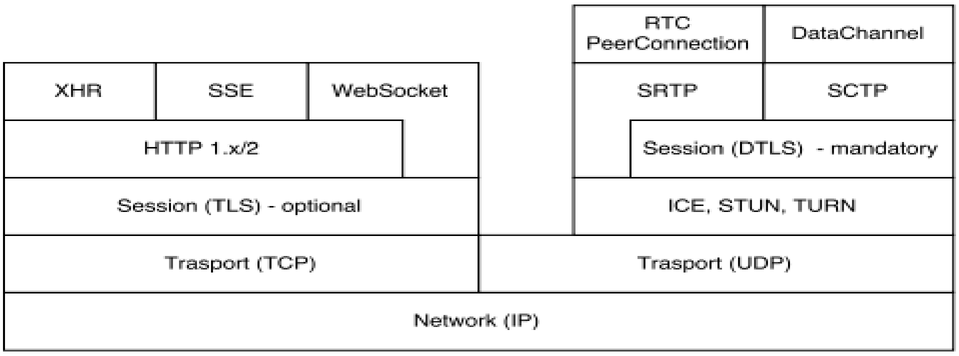

Figure 2.4.2-1 WebRTC Protocol stack

2.4.3 **Over the top (OTT)**

An over-the-top (OTT) application is any application or service that is provided over the Internet and bypasses traditional distribution or telecom infrastructure. Over the top services are most typically related to media and communication and are usually free or lower in price than the traditional method of delivery and not regulated.

The OTT applications has led to conflict of interest between companies that offer similar or overlapping services. The traditional ISPs and telecoms have to anticipate challenges related to firms that offer over-the-top applications. Consumers still pay the cable company for access to the Internet, but they might get rid of their cable package in favor of the cheaper streaming video over the Internet.

2.4.3.1 Telecom Fraud and Over-The-Top service

An OTT bypass attack includes a variety of redirecting the terminating traffic from legitimate mobile calls onto OTT applications. Recently it has become very difficult to detect the bypass fraud when the carriers use blending: only a part of traffic goes through the fraud channels. The revenue loss to the telecom operators, impacts of OTT bypass fraud include QoS degradation, charging of the receiver for the data service used, and inaccessibility of Value-Added Service (VAS) such as voice mail and call forwarding, customer dissatisfaction etc. [20]

Several countries are incorporating changes in their telecom regulations to address this bypass by issuing stricter guidelines for OTT players and interconnect service providers. As this is an ongoing process, telecoms remain vulnerable, bleeding revenue. Here is the quick summary of impact of OTT Bypass: Reduced Operators' call revenues Reduced Tax Revenues for Regulators Negative Customer Experience due to unexpected charges and quality of service issues Personal Security Issues Although, there have been a few approaches taken up by industry and vendors, there have been the following challenges to tackling OTT Bypass: Legal and Regulatory Complexity Market Risk Complexity Lack of Awareness/ Technical complexity Limited Vendor Solutions Fragmented Industry Approach Here are a couple of solution approaches that operators and vendors can take: a) Test Calling Based Solutions: This solution approach is inexpensive and easy to set up.

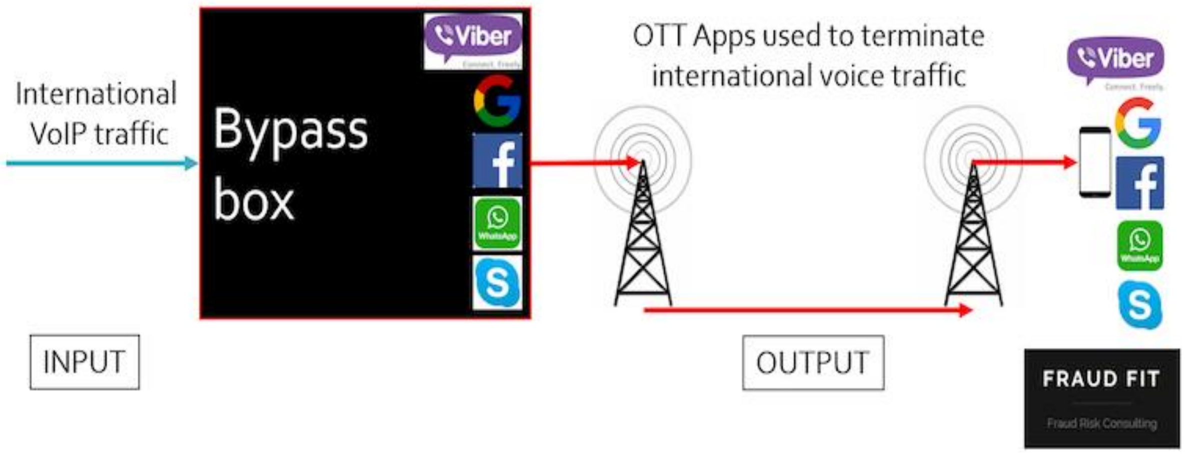

Figure 2.4.3-1 OTT bypass fraud scenario (Image source www.commsrisk.com)

General lack of awareness, an inconsistent approach by Industry groups & the complexities of the Legal & Regulatory aspects of OTT Bypass has slowed down the response to the issue. Operators, who have not considered this already, should undertake a fraud risk assessment, consider potential responses to OTT Bypass and take a suitable solution approach in order to take care of this issue.

2.5 Internet Service Provider network

The Internet consists of a worldwide string of connected networks that exchanges data through packet switching using the standardized Internet Protocol Suite (TCP/IP). Unlike before, people can stay at home and be connected to his or her family, friends, and even colleagues from anywhere around the world.

Internet communication is referred to as the sharing of information, ideas, or simply words over the World Wide Web, or the Internet.

Network challenges: ISP's has a hierarchical complex network divided into: -

Internet Gateway, backbone, core, edge and core-switching sites.

Internet Gateway network: An Internet gateway is a network "node" that connects two different networks that use different protocols (rules) for communicating in most case it's a gateway network that connects to different ISP's for worldwide web connection.

Backbone Network: A backbone is a part of network that interconnects various pieces of network, providing a path for the exchange of information between different virtual private network or subnetworks with a faster speed some terabytes per seconds.

Core Network: A core network is a telecommunication network's core part, which offers numerous services to the customers who are interconnected by the access network.

Edge Network: Edge network provides information exchange between the access network and the core network. The devices and facilities in the edge networks are switches, routers, routing switches, IADs and a variety of MAN/WAN devices, which are often called edge devices. Edge network provide entry points into carrier/service provider core/backbone networks.

Access Network: An access network is a type of telecommunications network which connects subscribers to their immediate service provider. It is contrasted with the core network, which connects local providers to one another.

ISP is literally a company that provides access to internet for virous organizations, business as well as individuals. According to[44] Internet architecture at the top Internet providers layer connected are called Tier 1 networks, usually are higher in traffic capacity and connects a various autonomous sytems and they don't pay to other similar tier 1 network, some Tier 1 included AT&T, Verizon, Sprint and etc.

Tier 2 ISP connected with a payment to tier 1 ISP and peers with other tier 2 ISP's in which provides a internet traffic to tier 3 ISP's. Tier 3 ISP are basically a stub network in which pays other ISP or transit network providers for internet connectivity.

ISP's besides providing access to internet provides domain name registration, web hosting, co-location(data centers for cloud computing), email and others. ISP's also provides connectivity other smaller internet sevice business organization usually called virtual internet service providers (vISP).

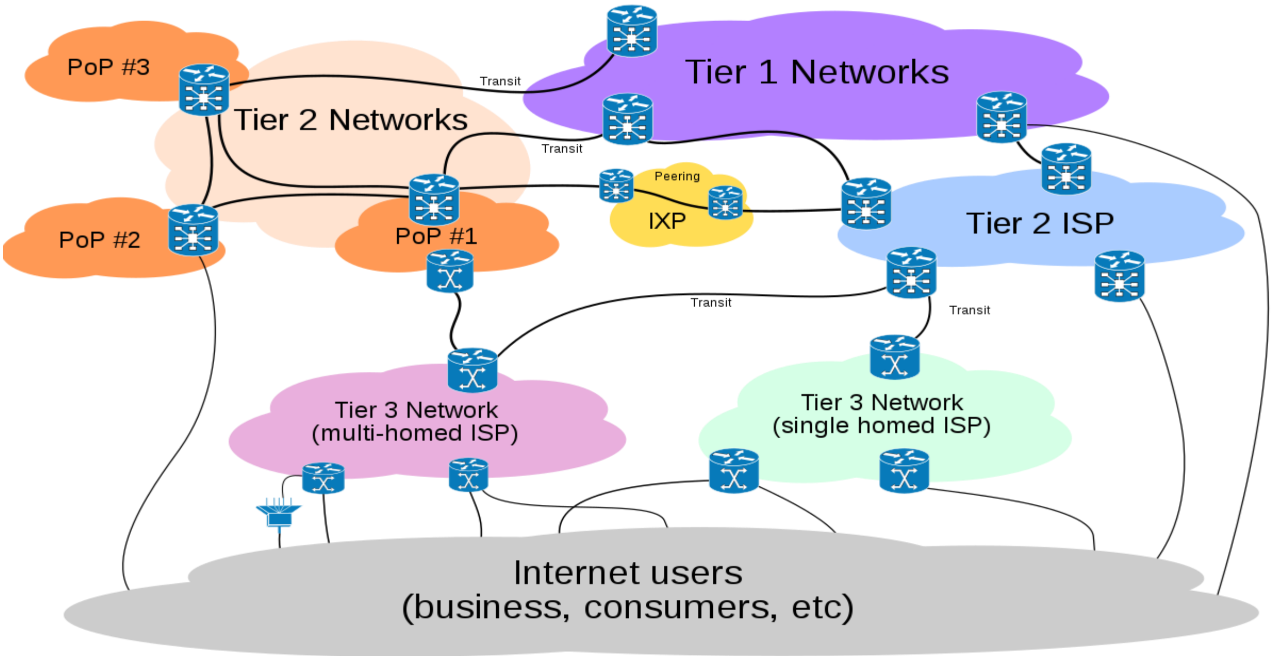

Figure 2.4.3-1 Internet and Various ISP connectivites(source [21])

Due to a hierarchical nature of the service provider network from provider access network to OSI layer 3 IP network first provider edge node the topology can be best represented in a fat-tree (Figure 3.2-1) in which show the network link increasing in bandwidth as we go higher to the internet gateways.

With the initial assumption of that a maximum customer that can be partitioned or configure in edge router with a CS(Core Switch) layer-2 device in our case will be 4094 by 4094 VLANs (i.e QinQ which is an amendment to the IEEE 802.1Q specification that provides the capability for multiple VLAN tags to be inserted into a single Ethernet frame) can be a huge amount of customer in which can be categorized into a single subnet 10.x.y.0 /16 subnet with max layer 3 host IP.

2.5.1 **Cloud Communications platform as a service**

Due to the rapid break through and technological innovations in telecommunications links such as satellite, fiber optics, wireless 3G,4G/LTE and now 5G's has increase bandwidth plus speed in which has led to introduction of various cloud service technologies. Cloud communication is one of the various types which is internet-based real time video, voice and data communication. Cloud communication computing architecture is composed of at least three layers: resource, platform and application that are hosted by usually a third-party or OTT service providers and can be access via public internet.

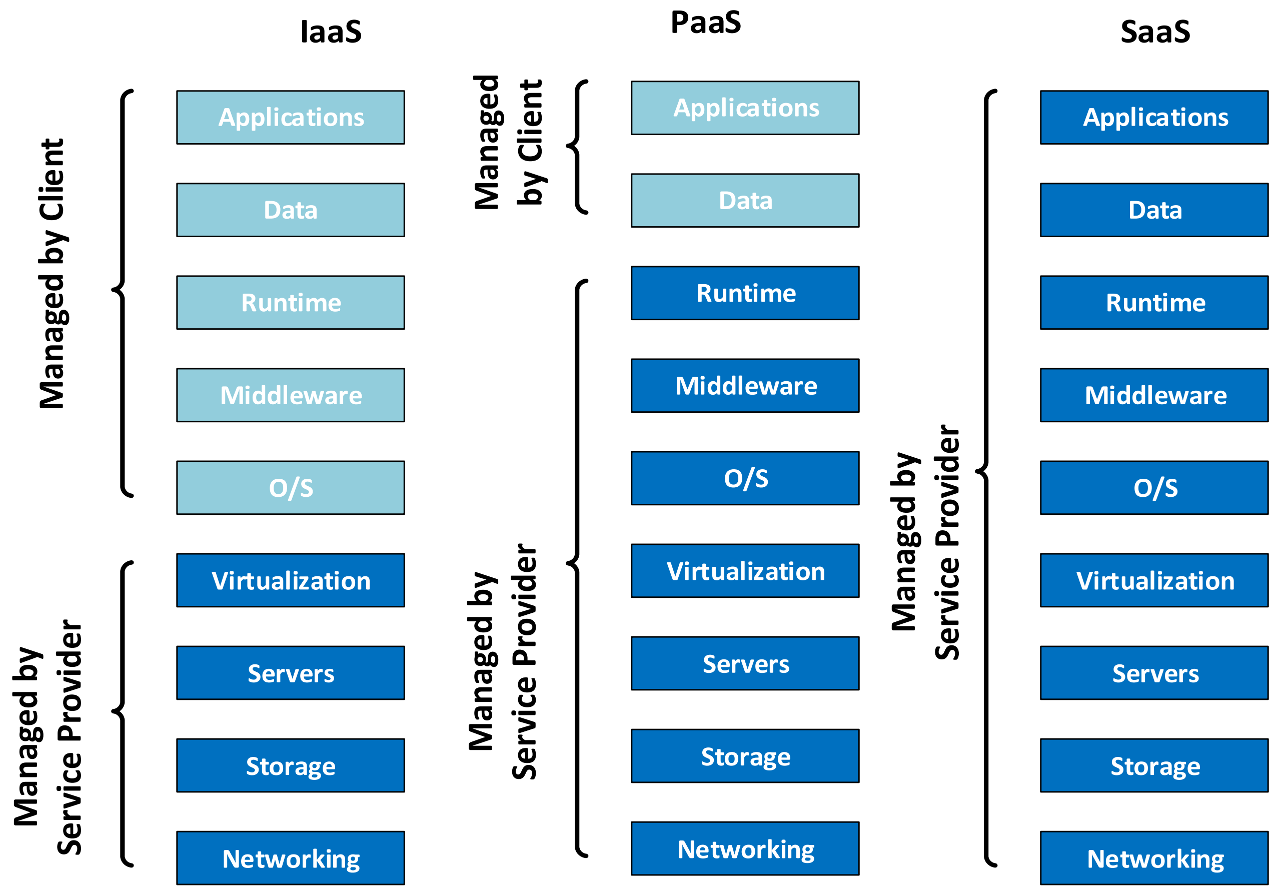

Figure 2.5.1-1 IaaS, PaaS & SaaS Cloud types management by provider and user

2.5.2 **Public vs Private vs Hybrid cloud services**

Public cloud services

Public clouds are a widely used and well-known cloud computing models in which usually open to or reacheablet to the public internet users. Some of the exmaples includes Amazon Web Services, Microsoft Azure, and Google Cloud Platform, in which clients data and activities are kept secured and isolated from others in a safe and integrated virtual space. Public cloud allows user to use by rent or susbcripiton the infrastructures and service provided ratather than owning and operating hardware by themselves. WebRTC cloud service can be implemented with other connetend delivery networks(CDN) and other legacy value added services(VAS) such as Web hosting, domain name service, public ip and other service provided by the service provider. [22]

Private cloud services

Private cloud services are as the name implies cloud computing services offered to only seleted users not to the general public over internet. Private clouds are usually called corporate or internal clouds, in which provide many benefits to the business.

Hybrid cloud services

A Hybrid cloud service is an environment that implements both private and public cloud services. Most business organizations are moving toward the idea that they need various types of cloud services in order to meet a variety of customer needs. [23]

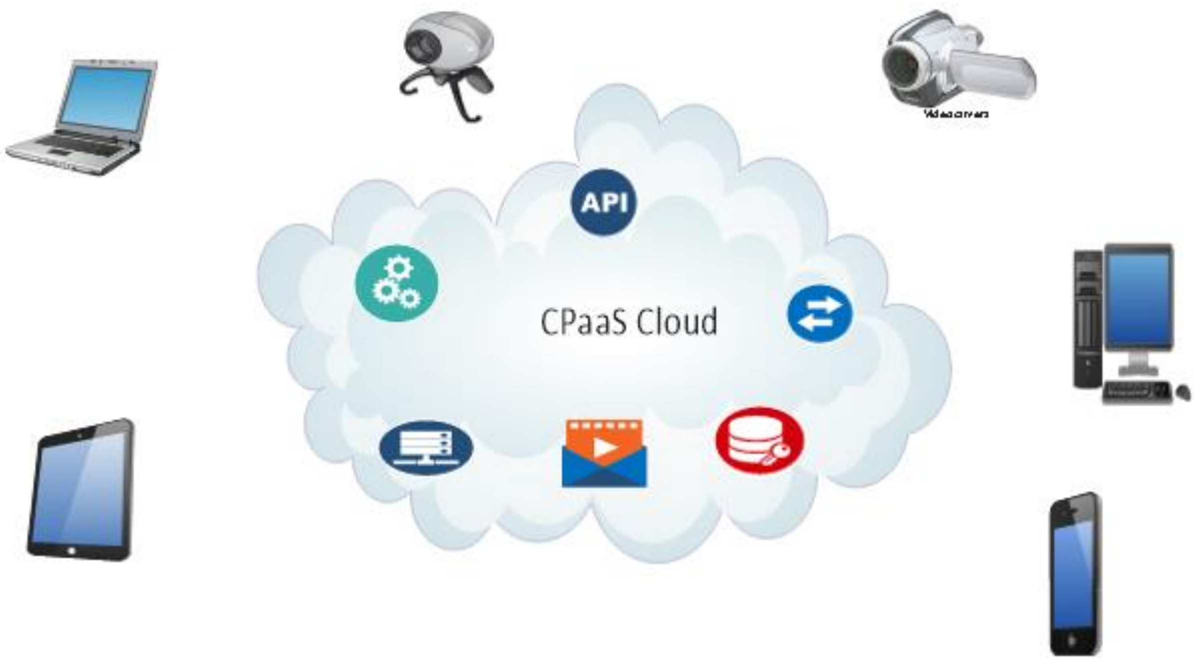

Figure 2.5.2-1 Communication platform as a service

2.5.3 Internet Network VoIP QoS analysis

For most service providers internet gateway routers are the last hop in the network autonomous system(AS) such that traffic management will be limited outside their AS or depends on the agreements between various ISP's. We will discuss later about what happens to a packet leaving the ISP internet gateway, but now focusing on a VoIP packet or IP communication inside the ISP, might have several hops to travel in order to pass the internet gateway and NAT translation too if the customer doesn't have a public IP address.

During troubleshooting a network connectivity problem or outage, checking network performance is necessary to determining weither the network is slow and what is the root cause (e.g., saturation, bandwidth outage, misconfiguration, network device defect, etc.).

The following performance measurement assumptions are key in-service delivery.

• The sent packet, packet loss and source to destination delay are taken from the measured statistics on at each end-device. This gives a proper estimate of end-to-end delay for the VoIP communications.

• A packet loss considered the difference between the number of packets sent at the traffic

• From source to destination packet delay statistics taken form the receiving clients.

• ICE route and signaling change effects will be assessed with respect to the bandwidth and number of hops from source to destination.

2.5.3.1 Delay

Voice packet total quality is a function of lots of components that includes the compression algorithm, delay, errors and frame loss, and echo cancellation. Accourding to International Telecommunication Union(ITU) consideration network delay of voice applications in recommendation G.114 defines three bands of a one-way delay depicted in the table below. [31]

Range in Milliseconds	Description
0-150	Acceptable for most user applications.

150-400	Acceptable provided that administrators are aware of the transmission time and the impact it has on the transmission quality of user applications.
Above 400	Unacceptable for general network planning purposes. However, it is recognized that in some exceptional cases this limit is exceeded.

Table 2.5.3-1 Delay spacifications for applications

There are two types of delay in voice one-way transmission called fixed and vaiable. Fixed delay elements are directly incremental to the overall delay on connection. But variable delays results from queuing delays in the input buffers on the interface connected to WAN, these buffers produce varriying delays, called jitter, over the network.

2.5.3.2 Network latency

A network latency is the mesure of delays in which a sender propagates packets to the destionation via various network elements. Latency is measured in milliseconds (ms) and is inevitable due to the way networks communicate or operates End-to End. [26] Network delay can be divide into

- Transmission delay: is also known as paketization delay is the time it takes to push all packets bi into a wire which is limted by the bandwidth(data-rate) of the link.
- Processing delay: It the time it take for a router to check the packet header and send it to the destination.
- Propagation delay: is the total time it takes for a packet traversal along the network path from source to destination.
- Queing delay:is the time a packet spent on a router queue to be processed or sent to the destination on exit interface.
- Bandwith limits: the maximum number of byte supported by the network QoS.

Network Latency caused by various transmission mediums can be calculated the Round Trip Time (RTT) from a packet sender to the reciever device or the Time to First Byte (TTFB): RTT is the value of a time it takes for a packet to get from the sender to the reciever and back. TTFB is amount of time it takes for the receiver to receive the first byte of data on the network when the sender sends the request [27]. The table below provides an illustration of the approach that can be used for delay specification:

	Less interactive application	Highly interactive application
Delay related to packet size, approx voice packet duration	20 mS packets → 20 mS	10 mS packets → 10 mS
Jitter buffer size, approx 2 x expected typical jitter level	20 mS jitter → 40 mS jitter buffer	20 mS jitter → 20 mS jitter buffer
Worst case jitter buffer size approx 2 x max jitter level	50 mS jitter → 100 mS jitter buffer	20 mS jitter → 40 mS jitter buffer
Expected typical IP network delay (one way)	50 mS typical	50 mS typical
Maximum IP network delay	200 mS maximum	100 mS maximum
Expected total one way delay	***20+40+50 = 110 mS***	***10+20+50 = 80 mS***
Max total one way delay	***20+100+200 = 320 mS***	***10+40+100 = 150 mS***
Worst case limit	***400 mS***	***250 mS***

Table 2.5.3-2 Delay for interactive applications

2.5.3.3 Netowrk Jitter

Network jitter is the hardest trap for a real time communication, streaminnig or real time applications over internet. A network jitter is a delay variation on incoming or outgoing packet that puts stress on the receiving endpoint, as it is trying to figure out the right sequence of data packets such a as UDP packets in most real time communications. Thus, since most real time communications are connectionless i.e there is no packet resend like TCP protocol this negative aspect leads to congestion and packet loss. Network jitter measured in milliseconds (ms), happen due to many reason such as interface or bandwith congestion or limitation, route changes, physical interfirance etc. Jitter can be measured by finding the average of the time difference between each packet sequence. Let pt={p1,p2,p3,p3 …. pn} be packet sent time in ms and total difference s={|p2-p1|,|p3-p2|, …|pn-p(n-1)|} , then total jitter

$$J_{total} = \sum s(pn\text{-}pn\text{-}1)/n/2 \quad \text{……………………….(Equation 2.5.3-1)}$$

For service offered out of the ISP such as OTT real time communications it can be classified into two jitter inside ISP and outside network can be transonic or satellite and multiple ISP's. Therefore, total jitter for user accessing an external network service average jitter calculated,

$$J_{total} = \frac{1}{2}(J_{isp} + J_{extisp}) \quad \text{…………………….. (Equation 2.5.3-2)}$$

2.5.3.4 VoIP Internet WebRTC signaling analysis

The ISP network connecting various VoIP OTT service providers to the users as show in figure 3.4.1. Internet users from the service provider network can take different path to the same or different OTT service provider through different internet gateway links. Service providers can have multiple links to internet in which connected via transonic or satellites usally called an internet exchange point (IXP's). IXP is a physical infrastructure which Internet service providers and content delivery networks(CDNs) exchange internet traffics between their network's or autonomous systems [24].

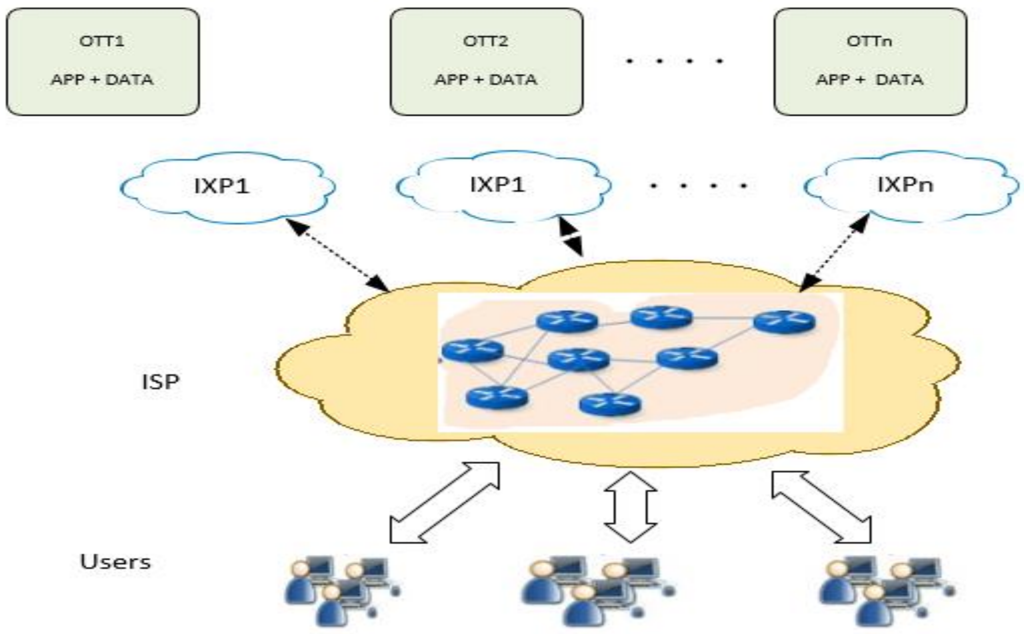

Figure 2.5.3-1 Users connecting to various OTT service providers

Internete service providers connect end usesr business to internet through various options such as a transit ISP peer or IXP.

Service provider network usually modeled with a three level herarchy, i.e access, distribution and core network but in some case if the service provider span cover a large area such as nation wide like ethiotelecom the hierarchy depth increases such that a backbone network will be on the top of access, distribution, core neworks. All clients are connected to the access network or provider edge using ip address provided by the ISP, access networks are connected to the distribution netwok where packets generated at users are fast switched using technologies such as IP over MPLS(multi-protocol label switching). Distribution network in turn connected to core network, in which core network aggregates access networks and have higher capacity.

Sence internet or as we call it world wide web(www) is in essence a network of networks with a various types of data traffics and set of protocols that flows on such that it is owned by all or no one but usually ISP,CDN and IXP’s tends to cooperate. So that it is concincing that when we use a VoIP over internet the application use a ‘best effort’ class of delivery because in the internet, real-time voice and video packets are mixed and contending with all other forms of traffic such as web,ftp, and email. Therefore, it can be said that it is inconceivable to promise a real-time voice and video quality across an internet connection as a service provider because public internet does not support QoS.

2.5.4 **Related Work**

In recent years, WebRTC Cloud real time web application service providers such as Google, Twilio, Tokbox, Kandy and many others, have offered an evolving web communication services. Most web browsers are now capable of WebRTC technology in which the securely get local media input devices audio and webcam video sources and stream it to the destination party and also get remote stream audio and video present it in HTML5 web page.

In [24] presented design of on demand a NaaS (Network as a service) architecture such that application service providers(OTT) and application service themselves request to request specific network services from a network service provider within a NaaS. NaaS architecture on demand network function feasibility shown with SDN-enabled NFV architecture presented in which have gaps in showing detail part of service provider role.

NUMBMEDIA[25] proposes open source WebRTC as a PaaS general focus on the development and scalablility of WebRTC PaaS. This open source implementation steel relies on software based best effort routing and doesn’t consider network infrustrature.

In [26] suggest a specialized network service for WebRTC in which is a turn based architecture since WebRTC uses a best-effort routing but it would benefit from better quality and specialized network services provided by network operators or telecoms. A depicted in the figure below exemplary network connection a better network service is available for eligible users from the network service provider allowing a special network service.

Author	Title	Critical Review
Amina Boubendir, Emmanuel Bertin No¨emie Simoni	On-demand Dynamic Network Service Deployment over NaaS Architecture	Scope limited to deployment models with a WebRTC OTT application and showcase the on-demand dynamic deployment of virtual network functions, mainly TURN server. Doesn't provide WebRTC business solution for cloud customer request.
Boni García Luis López Francisco Gortázar	NUBOMEDIA: The First Open Source WebRTC PaaS	Focus on cloud API for WebRTC from the deployment perspective for open source WebRTC PaaS service. Doesn't present a network infrastructure or network service provider network role.
Vamis Xhagjika O` scar Divorra Escoda Leandro Navarro Vladimir Vlassov	Load and Video Performance Patterns of a Cloud Based WebRTC Architecture	Present joint factors that including multi- cloud distribution, network performance, media parameters and back-end resource loads, in Cloud based Media Selective Forwarding Units for WebRTC infrastructures.
Ewa Janczukowicz Arnaud Braud Stéphane Tuffin Gaël Fromentoux Ahmed Bouabdallah Jean-Marie Bonnin	Specialized network services for WebRTC TURN-based architecture proposal	Proposed a specialized network services for WebRTC, discuses about a technical and business challenges for network operators and WebRTC communication service providers to develop cooperative solutions.[27]
NIKOLAOS PINIKAS	A WEBRTC BASED PLATFORM FOR SYNCHRONOUS ONLINE COLLABORATION AND SCREEN CASTING	Discuss about capabilities of the WebRTC APIs to implement a platform for synchronous online collaboration, screen casting and simultaneous multimedia communication by utilizing the WebRTC data and media streams.
Duong Tuan Nguyen, Kim Khoa Nguyen, Saida Khazri, Mohamed Cheriet	Real-Time Optimized NFV Architecture for Internetworking WebRTC and IMS	Propose NFV architecture to enable effective communication between IMS and WebRTC users supporting smart community services. optimization model to design and allocate resources for such system while ensuring desired QoS level.
Pelayo Nu˜no, Francisco G. Bulnes, Juan C. Granda, Francisco J. Su´arez, Daniel F. Garc´ıa	A Scalable WebRTC Platform based on Open Technologies	A limited view on scalability from as signaling server perspective only.

Amina Boubendir Emmanuel Bertin No¨emie Simoni	Network as-a-Service: the WebRTC case How SDN & NFV set a solid Telco-OTT groundwork	Present challenges made by OTT on telecom service providers and suggest a cooperation between OTT providers and Network service providers to reduce the best-effort route and increase specialized network service.

Table 2.5.4-1 Related works .

CHAPTER 3

Proposed WebRTC CPaaS Model

Proposed WebRTC CPaaS cloud architecture, WebRTC VoIP signaling and media design and implementation on the provider network as shown in the figure 3.1-1.

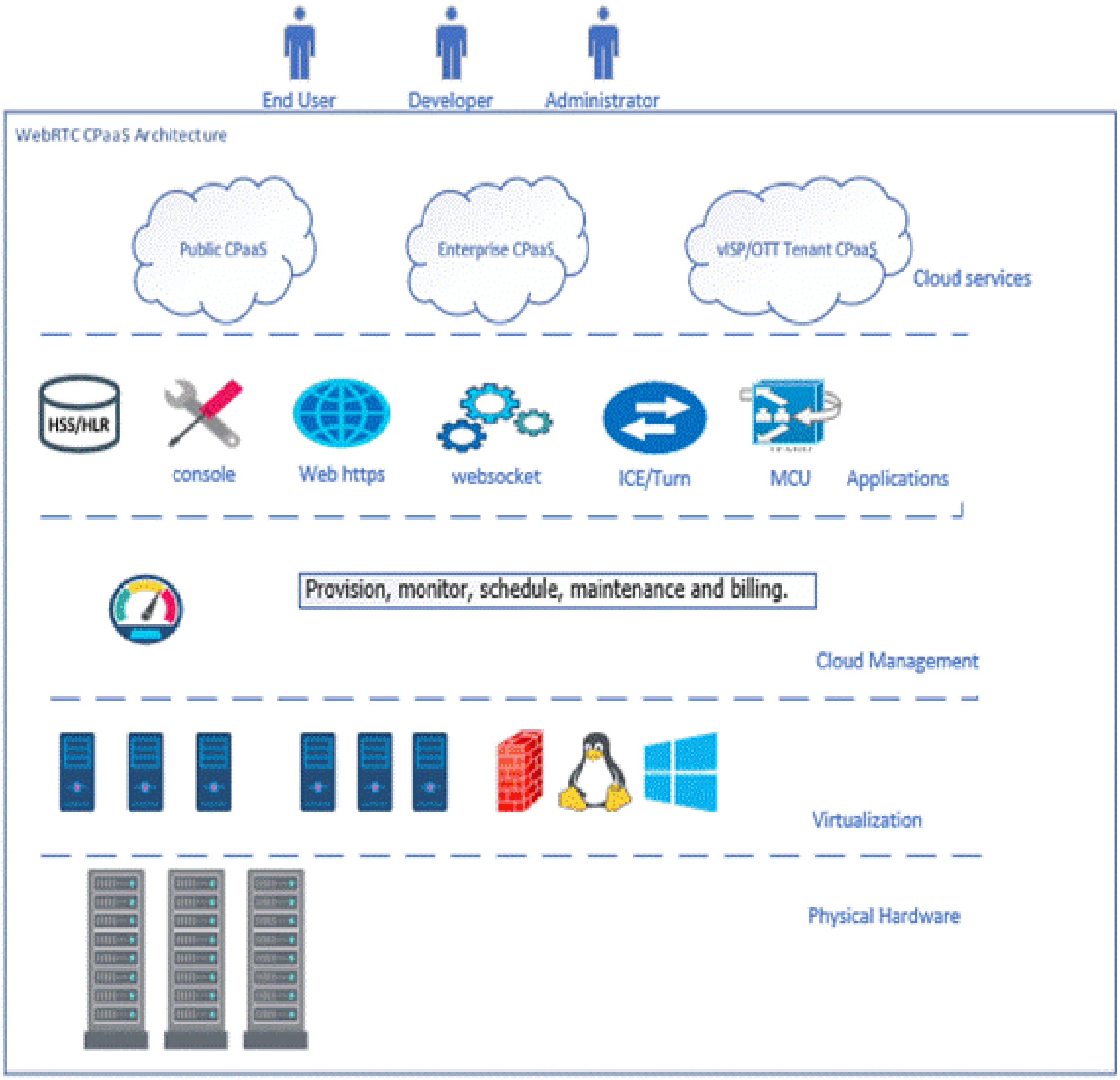

Figure 3.1 1 WebRTC CPaaS Architecture

The proposed RTC can be segmented into two categoris:

a) WebRTC Cloud Platform Architecture

b) WebRTC Application & Signaling Gateway Architechtecture,

3.1 WebRTC CPaaS Architecture overview

In order to mitigate current and future OTT services as well as VoIP frauds on ISP this research propose a WebRTC CPaaS solution as well as ICE STUN and TURN signaling services offered by the service provider.

The WebRTC Cloud CPaaS can be modeled as shown in the figure 3.1-1. As discussed in section 2.5 most service providers have at most access provider Edge router (PE), and core (CR) and backbone (BR) with internet gateway router (IGW) network parts that interconnects the entire ISP clients based on the size of the telecom ISP.

As present days techlogocial requirements with evolve business needs and various types of contents as well as communication with a high speed, the CPaaS design should overcome obstacles such as geographical locations, system fail over recovery time and high-availablity issues.

Since the network infrusture is owned by the service provider implementation of signaling will be easy both for public and private network domains if provided by the service provider. However, the proposed solution architecture is open to the option that clients can use their own signaling application server. CPaaS solution provider with the assumption prepares data center server farm with DCN(Data Center Network) each server hardware is used for various OS platforms.

This CPaaS Architecture designed to solve the process of putting the best possible WebRTC core architecture in use by cloud service users and easily integrate with existing legacy services and IMS. Also provides flexible environment for management by system admin and easy platform for end users developing, launching and maintaining any WebRTC applications.

CPaaS Data centers(DC) can be distributed over a large geographical areas throughout the provider network and conneted via a high speed network. A nation wide telecom service provider like ethiotelecom might need a CPaaS data center at each major cities or aggregation or edge site. CPaaS extends the principle of Plaltform as a service by adding a communications futures thus end-to-end best network quality will be the primary demand.

The Data Center resource for the CPaaS can be devided into three separate resources:

- Cloud Infrastructure for CPaaS: includes Networking, Storage, Hardware, hypervisor(Virtualization)
- Cloud Platform runtime : OS and middleware, Network services

- Cloud Data center network vs consumer network: DCN for Servers and communication/signaling

3.1.1 **Cloud Infrastructure for PaaS**

The proposed solution for CPaaS infrastructure can be implemented inside existing standard data centers with existing customer networks such as web hosting, domain service, and other co-located services. For example, web hosting customers can add real-time communication feature to there pages by easily integrating to CPaaS service.

Since the delivery model of CPaaS is related to PaaS, i.e. instead of providing only the runtime over the internet, CPaaS provides a platform for communication software creation without worriying about hardware and Operating system. There are two choice for deploying multiple isolated services on a signle platform containers and virtual machines.

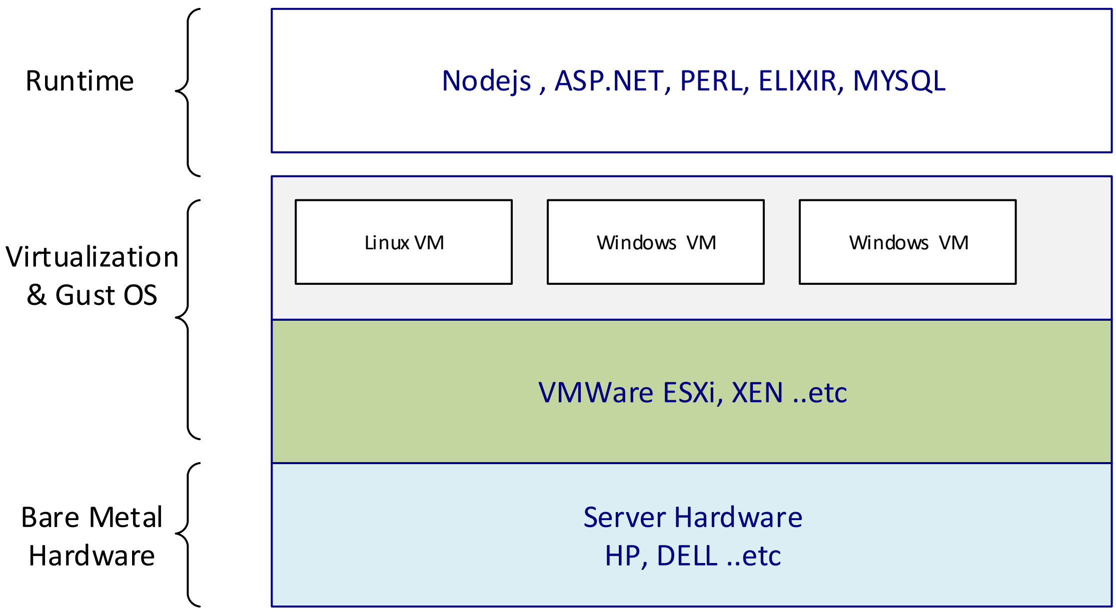

Figure 3.1.1-1 CPaaS cloud for a type-1 hypervisor layers

3.1.2 **Cloud Platform Runtime for CPaaS**

The cloud runtime holds various various operating systems based on the choice of the service provider and the cloud client. User can decide to run in a hypervised or doker container to isolate their application and its dependancies into a self-contained system. Runtime environment usually is an execution platrform that allows user application to interact data. Runtime environment requires storage, memory and CPU resource allocation.

Application development programs usually include a runtime enviroment component that helps the developer to run the application test and debug. Thus, CPaaS provider must provide all available runtime environment as a choice independent of the OS.

3.1.3 Cloud DCN and Consumer network

A provider edge router(PE) is a network element that holds broad range of residential, buness enterprise and infrastructures services and applications. PE routers holds a range of protocols such as BGP(Boarder Gateway protocol), OSPF(open shortest path first) or IS-IS (Intermediate System - Intermediate System) protocol for edge router and core routers peer connection, and Multiprotocol Label Switching (MPLS) for forom PE to provider CR router communication.

3.2 WebRTC Signaling and Gateway

Signaling is the primary and basic steps towards communication i.e. it is the process of coordinating communication in point to point or point to multi-point direction. A Simple WebRTC application to set up a peer to peer call, both terminal points need to exchange the following information:

- Session control messages used to open or close communication.
- Error messages.
- Media metadata such as codecs and codec settings, bandwidth and media types.
- Key data, used to establish secure connections.

Proposed WebRTC signaling and Gateway is distributed or implemented at each provider edge site such that each ingress traffics within the provider edge will be redirected with Geo-DNS to the nearest STUN/TURN server for communication signaling and relay. These implementation of STUN/TURN server close to the user end will rediuce route discovery latency and peer to peer signaling fast.

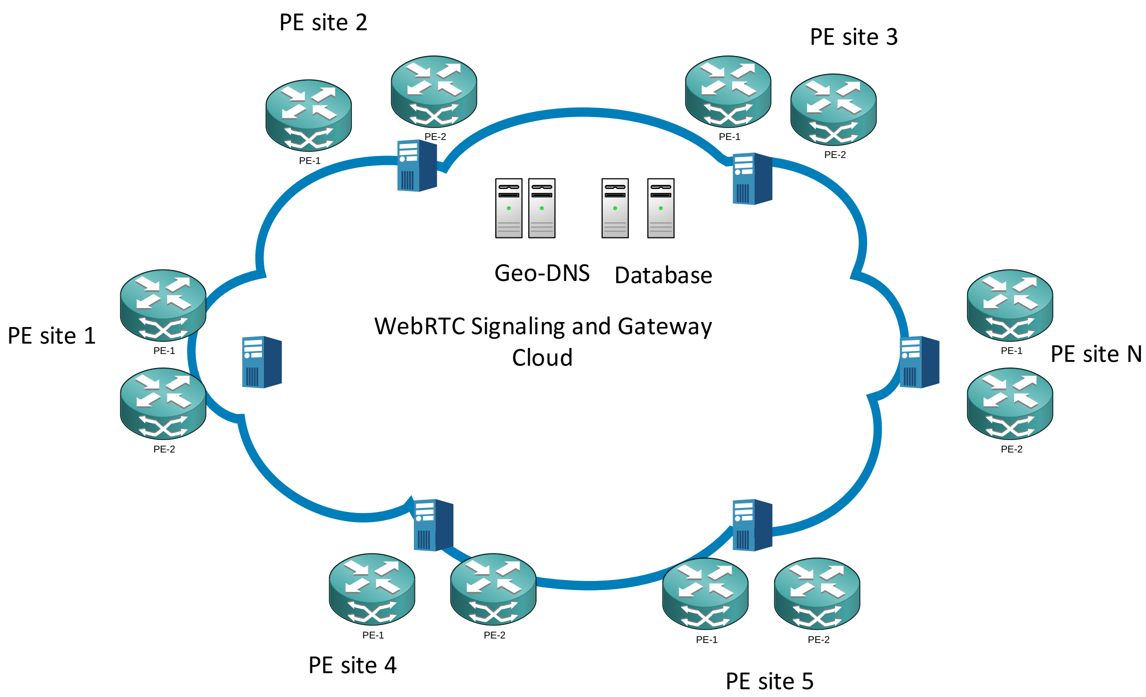

Figure 3.1.3-1 WebRTC Signaling and Gateway

3.2.1 ICE STUN/TURN Server

ICE STUN/TURN server allows candidates to represent connections over either TCP or UDP, with UDP usually selected (and being more generally supported). Each protocol supports a small types of candidate, with the candidate types defining how the data makes its way from peer to peer across any network.

ICE candidate selection

When two users wants to communitcate using WebRTC, they need to exchange connectivity and media informations. Due to various types network conditions, such as hosts residing behind Firewalls/Routers with one of the four types of NAT(Full cone, Restricted cone, Port restricted cone or symmetric NAT). Thus, the ICE STUN protocol used to exchange information in which a host with a private IP addresss needs allocation of public IP address and PORT. After gaining the host candidate from its local IP address interfaces as figure 3.2.1 shows, when clinet1 browser sends a STUN binding request packet to get a reflexive candidate as shown(messages 1 to 3) to a STUN server. The NAT on the local router creates a binding for the request that becomes the server reflexive candidate for RTP. Using the server reflexive candidate client1 browser sends an SDP offer message to client2 browser (message 4). In turn user agent 2 by collecting host candidate from local IP address proceeds to obtain a server reflexive candidate (messages 5 and 7), which is not identical to its host candidate because it is behind another NAT router. The duplicated candidate is discarded leaving only the host candidates.

Because user agent 1 started the communication it is selected as controlling and user agent 2 is will be a controlled. User agent 2 tries a connectivity start but since it is controlled it does not have a valid attributes to reach user agent 1 through the NAT so the request is dropped (message 8). User agent 1 being the controlling participant has the attribute to traverse the NAT device with its aggressive nomination STUN connectivity check (messages 9 to 12). After getting the STUN binding request with aggressive nomination user agent 2 willl make a matching check using the attribute from user agent 1's STUN binding request to confirm the connection (messages 13 to 15). At this point both users agent's have verified that the connection is valid and it has been nominated for use for this media stream. Now the two peers regardless of their network topologies and ip address can now send media through this connection. Listed code below is a typical stun server configuration in WebRTC

```
  const configs = {'iceServers': [{'urls': 'stun:stun.l.google.com:19302'},
{'urls': 'stun:test.ethio.edu.et:3478'} ]}
    const peerConn = new RTCPeerConnection(configs);
```

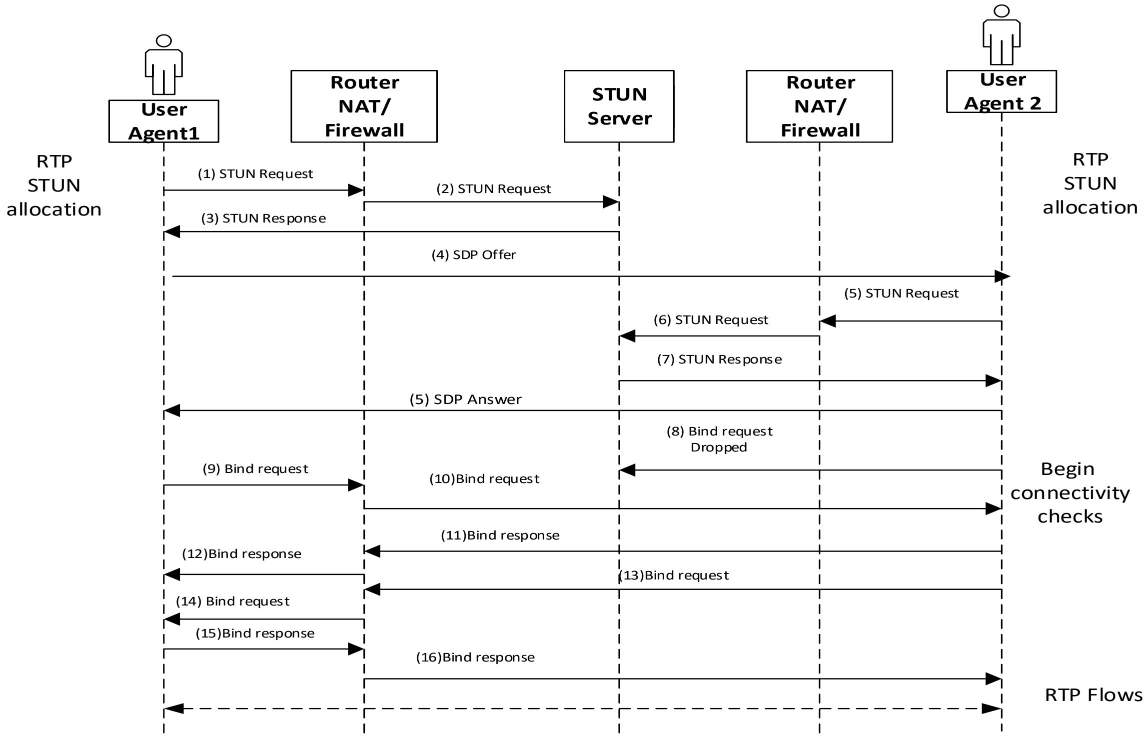

Figure 3.2.1-1 ICE communication flow using STUN

When a direct communication fails between two WebRTC end points using stun protocol across a NAT router or firewall due to symmetric NAT or firewall policies an extension of STUN protocol TURN will be a choice in relaying RTP media. Listed code below is STUN alternative TURN configuration.

```
const iceConfiguration = {
    iceServers: [
        {
            urls: 'turn:test.example.com:19403',
            username: 'username',
            credentials: 'auth-token'
        },
        {
            urls: 'turn:test.example.com:5349',
            username: 'username',
            credentials: 'password'
        }    ]
}

const peerConnection = new RTCPeerConnection(iceConfiguration);
```

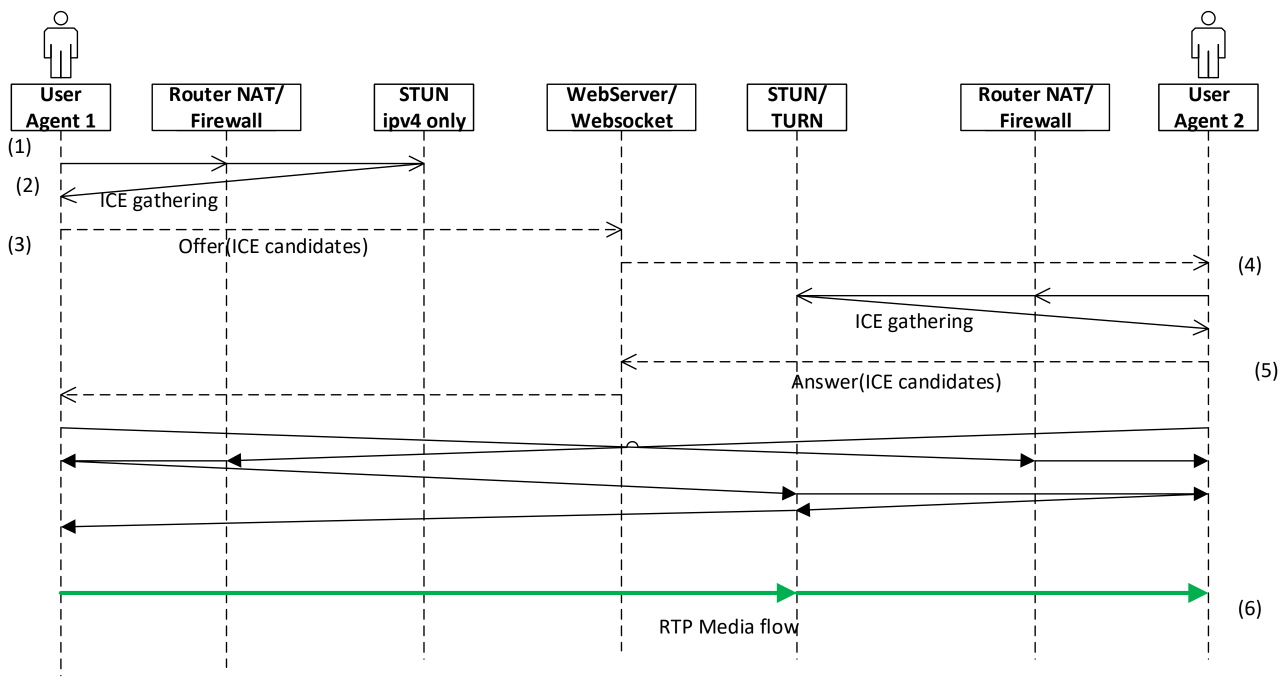

Figure 3.2.1-2 TURN server communication

From telecom service provider infrastructure perspective cloud communication implementation is very easy since most of the network and data center network are already deployed. Distributed cloud implementation is a cloud communication that has geographically placed infrastructure that primarily runs services at different location but runs like one service as a whole by using a load balance and clustering technology. This is different from the theoretical cloud model that relies on a centralized data center.

3.3 Algorithm

This algorithm is developed specifically for WebRTC App and CPaaS cloud service that a service provider can give a quality VoIP service with WebRTC technology infrastructure. Such that in addition to the traditional PaaS cloud orchestration and management additional communication future will be added to cloud resource. Today, there are many free opensource and commercial stacks available for IaaS and PaaS despite they don't usually fully developed to satisfy the user but they allow the user to add his own plugins, widgets, custome pages and etc.

3.3.1 New CPaaS provisioning resource allocation

In this allocation module just like existing ISP provisioning such as web hosting, domain name, mail box service, and other value added services cloud service for CPaaS is provisioned. Based on user enterprise scale a various data center distributed inside ISP network hierarchy.

Step in cloud resource allocation

Cloud user provides request to have virtual data center/s across the ISP network i.e. at different PE site

Cloud administrator get the request and check if any pre-request and check resource availability at various PE site.

Accourding to resource availability information at each mentioned datacenter where physical hardware servers are install PE site a hypervisor releases a processing elements(PES) for virtual machines(VM's) for the user WebRTC cloudlets to be installed.

A broker on the behalf of the cloud customer will allocate the processing of VM's and cloudlets processing with a process scheduling algorithm.

Algorithm 1: New CPaaS provisioning algorithm resource allocation

Input: Cloud user provides information for CPaaS cloud int Datacenter_id, VM's, Cloudlets, BW, RAM and Storage

Output: Resource allocation at each Datacenter done by cloud provisioning administrator

1: start

2: get request input (Datacenter_id, VM's Cloudlets, BW,RAM & Storage)

```
    while (request > 0)
      if  (check Datacenter_id) ← true
        if ( if resource available) ← true
          datacenter → initialize broker
          get Host, VMs, Cloudlet
          select VM allocation policy( space_shared, time_shared)
          broker → get VM list
          broker → get  cloudlist
          host Pe allocate for VM's
          broker install cloudlet in VMs
        end if
      else if
      return
      end if
    end while
create  database
add data, domain
stop
```

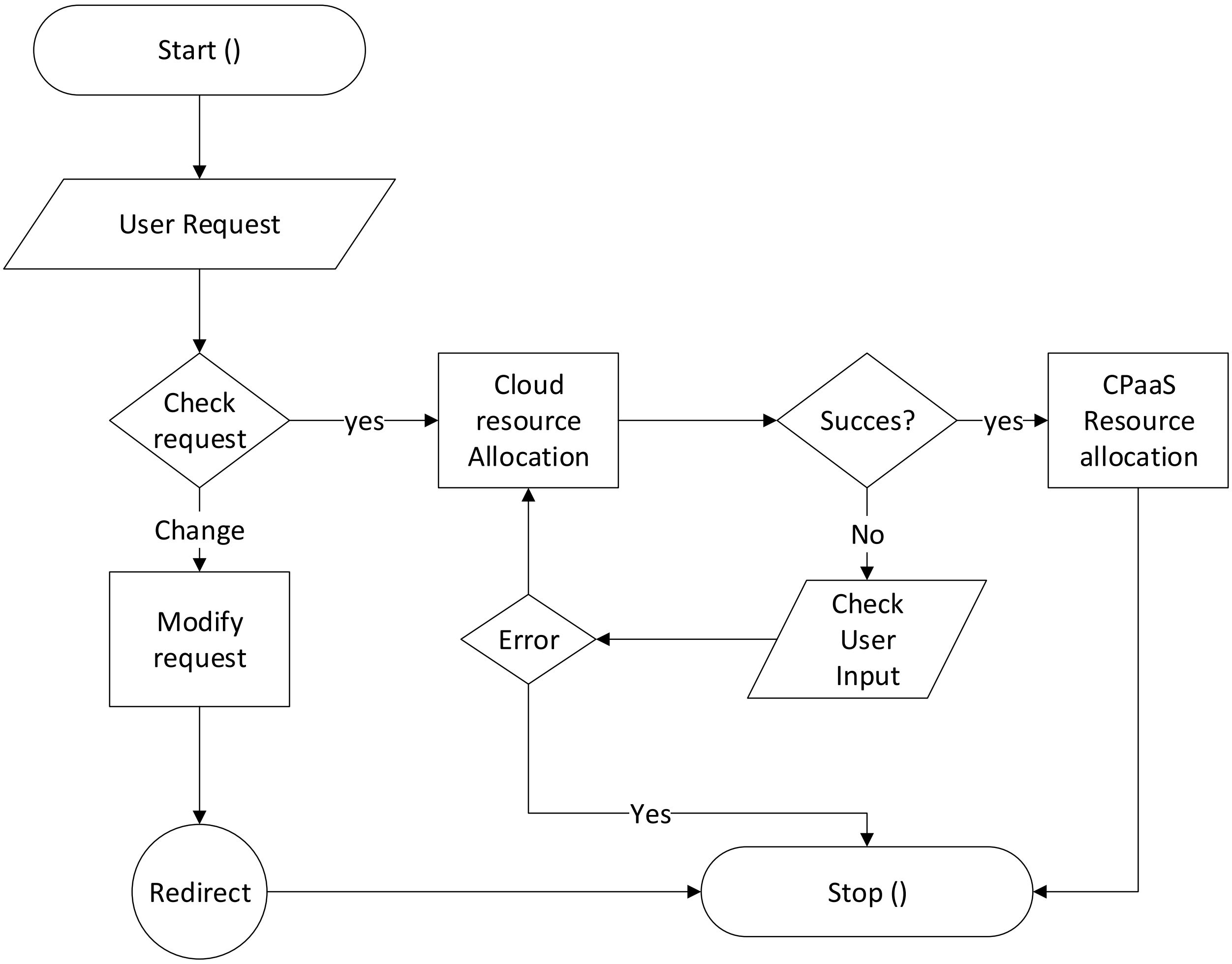

Figure 3.3.1-1 New CPaaS request process flow

3.3.2 Scaling CPaaS provisioning resource

Usually operational activities includes scaling of cloud resources depending on the business requirements such that excess resource will be downgraded and resource that are required are upgraded.

In a vertical scaling additional CPU, memory, or input output resources will be allocated to an existing provisioned resource, or changing exitsting server with a latest powerful server. Cloud administrator will check resource availability and adds the requested resources.

However, in horizontal scaling based on customer needs to meet their business demands, service provider will provision additional servers.

Algorithm 2: Scaling CPaaS provisioning algorithm

Input: Cloud user provides scaling information regarding int Datacenter_id, VM’s, Cloudlets, BW, RAM and Storage

Output: Resource allocation change based on existing information

```
start
get request input (Datacenter_id, VM’s Cloudlets, BW,RAM & Storage)
    while (request > 0)
        if (check Datacenter_id) ← true
            if ( if resource available) ← true
                datacenter → initialize broker
                get Host, VMs, Cloudlet
                select scaling type (horizontal, vertical)
                broker → add VM list
                broker → add cloudlist
                host Pe allocation for VM’s
                broker install cloudlet in VMs
            end if
          else if
          return
        end if
    end while
stop
```

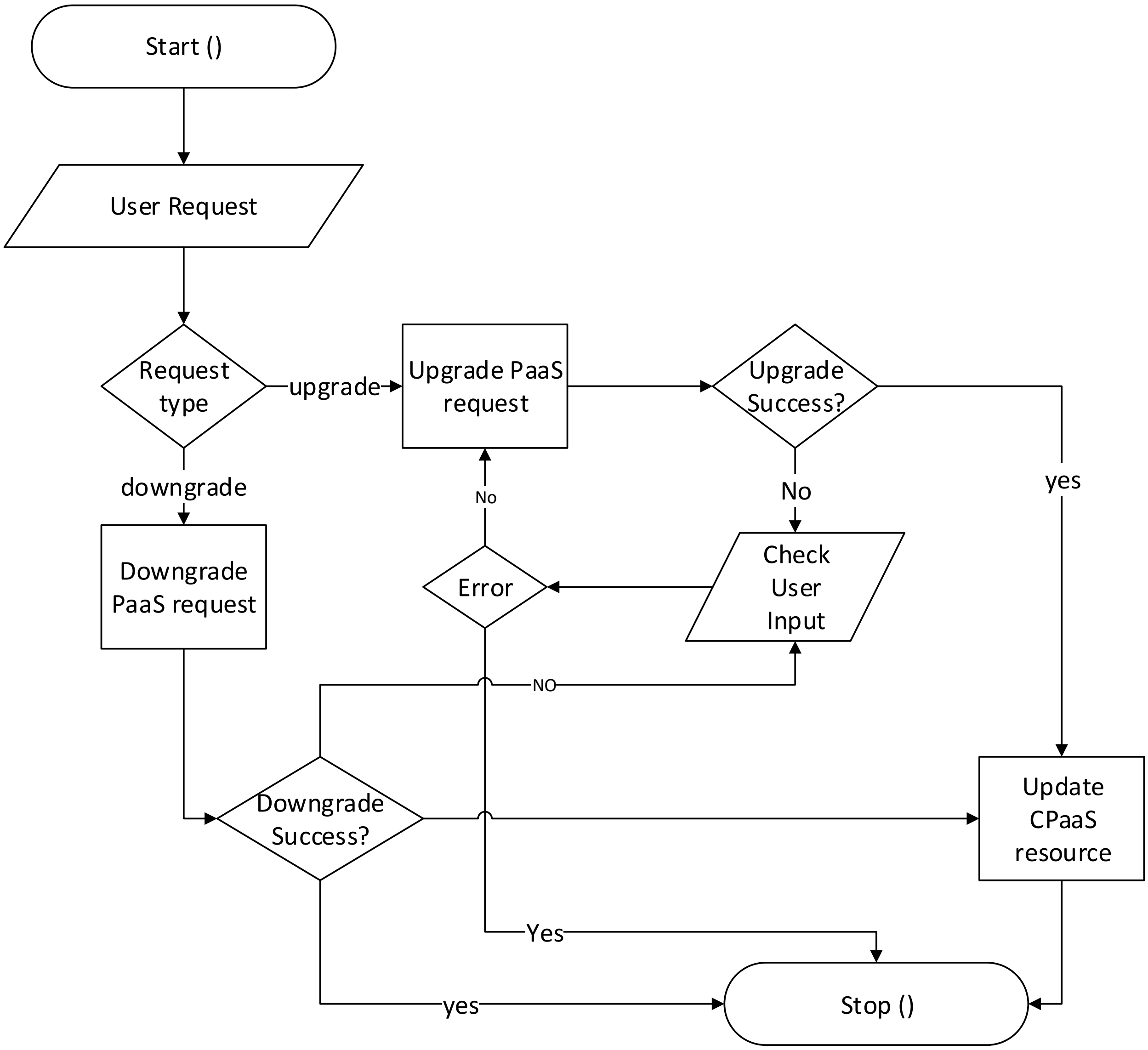

Figure 3.3.2-1 CPaaS Scaling upgrade downgrade request handler flow chart

3.3.3 **Terminiation of cloud resource**

Like any other cloud services agrments comprises multiple parts, the Terms of Service (ToS) - which includes termination and Service Level Agreement (SLA). Once the user request for service cancelation the cloud administrator will proceed in removing resourcces allocated and add it to the cloud resource.

Algorithm 3 Termination CPaaS provisioning algorithm

```
Input:  Cloud user provides termination information int Datacenter_id,
Output: Resource allocation at each Datacenter
start
get request input (Datacenter_id)
    while (request > 0)
        if  (check Datacenter_id) ← true
            if ( if resource available) ← true
                datacenter → initialize broker
                get Host, VMs, Cloudlet list
                broker → delete VM list
                broker → delete cloudlist
                end if
                else if
                return
            end if
        end while
    stop
```

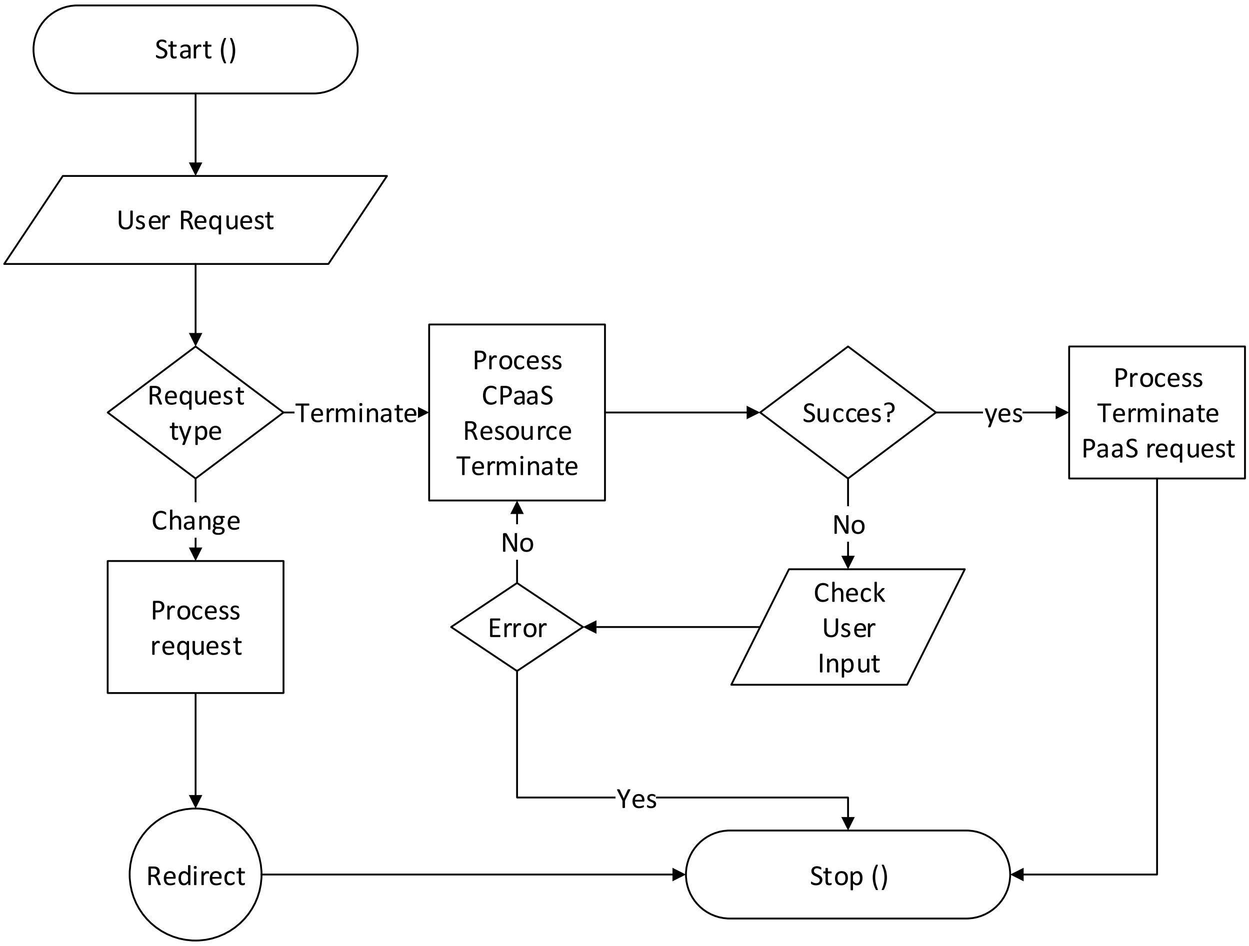

Figure 3.3.3-1 CPaaS Termination Request Handler flow chart

3.4 Cloudsim Plus Cloud simulator

A cloudsim plus is a toolkit(library) for pursuing software engineering methodoly for a better modularity, extensibility and correctness. Based on the CloudSim framework, it's goal is to improve several engineering aspects, such as maintainability, reusability and extensibility of a cloud service. Cloudsim Plus supports modeling of a large scale cloud service. [52]

The main components of cloudsim plus

Regions: It models geographical areas in which the service provider provisions cloud resources to their customers.

Datacenters: Models infrastructure service provided by the service provider. It consistes of a collection of computing host or servers that are either homogenous or hetrogenous in nature, base on their hardware configuration.

Datacenter: characteristics: it models information regarding data center resource configfruations as well as network setup.

Hosts: Models physical resources (computer or storage)

The user base: It models a group of users considerd as a single unit in the simulation, and its main resposiblity is to generate traffic for the simulation.

Coudlet: is job submitted in cloudsim plus, a small scale datacenter minimizes the latency usually placed closer to the end user in reaching the cloud servers and also resides close the portable devices.

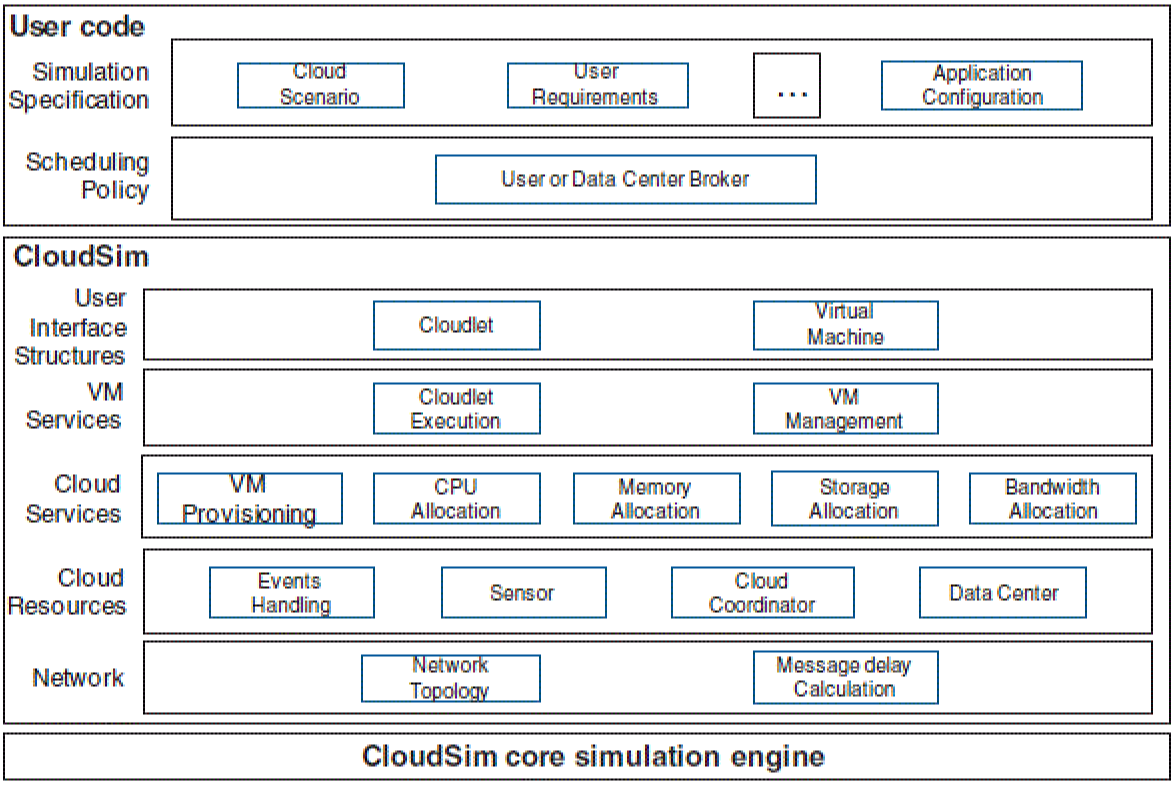

Figure 3.3.3-1 CloudSim simulation engine.

CHAPTER 4

Experiment Setup and Evaluation

4.1 Overview

This chapter discuss the evaluation step of the solution in detail, tools and technologies used in evaluation of WebRTC app and Signaling gateway test cases which can show the main functionalities of the solution are also discussed. In addition, the evaluation setup and considerations taken during the evaluation, result and discussion part is presented.

4.2 Hardware

In this experimental setup a highend laptop computers are used capable of high definition video and audio with fast codic capability.

Laptop 1 for WebRTC Client Test

Dell Latitude E5430

2 Physical Cores @ CPU 2.90 GHz (Intel® Core(TM) i5-3380M)

12 GB of DDR3 RAM

500 GB of SSD (1 × 500 GB)

1Gbps Network (1 × 1Gbps w/ TLB)+Wifi

Laptop 2 WebRTC Client Test

HP ELITEBOOK

4 Physical Cores @ CPU 1.90 GHz (Intel® Core(TM) i7-8665U-CPU)

8 GB of DDR3 RAM

500 GB of SSD (1 × 500 GB)

1Gbps Network (1 × 1Gbps)+Wifi

4.3 Software

Operating System

Ubuntu and Windows 10.

Ubuntu: is the world's most popular open-source desktop operating system, and we think this is our best release to date. Ubuntu latest version 20.04 LTS desktop best suited for enterprise and personal users.

Node.js: JavaScript runtime environment is an open-source, cross-platform, that executes JavaScript code outside a web browser fastly. Node.js runtime allows programmers to use JavaScript to write command line tools and for client sind and server-side scripting running to produce dynamic web application page before the web page application is opend by the user.

Coturn Server: is a free and open-source TURN and STUN server for VoIP and WebRTC.

MS Visual Studio Code: is best and free code editor bye Microsoft redefined and optimized for developing and maintaining latest cloud and web applications.

Eclipse : for Java Cloudsim Plus cloud simulation software.

Mysql-server with workbench community edition.

Browsers: Mozilla Firefox and Goolge Chrome.

4.3.1 WebRTC App User interface video GUI

In this part we will design and build a dynamic web client application that allows two users on separate devices to communicate using WebRTC. Our application will have user authentication login page with mysql as a database server and the other for graphical user interface for calling the other user for this we will use HTML(Hypertext Mark up Language) and Javascript.

Our web application will run on nodejs now a days traditional web hosting are no more optimal choice due to the preference of dynamic pages. HTML is the most commonly used markup language – a system designed to process, define, and present text by embedding tag and text annotations including styled files to make text manipulation easy for the computer. HTML5 if the fifth HTML version that supports virous virtual vector graphics, SVG (Scalable Vector Graphics), and canvas. In HTMLfeatures like vector graphics needs additional different technologies like Silverlight, Flash plugins, or VML (Vector Markup Language).

Our demo WebRTC peer to peer application server and client side code structure is presented as below.

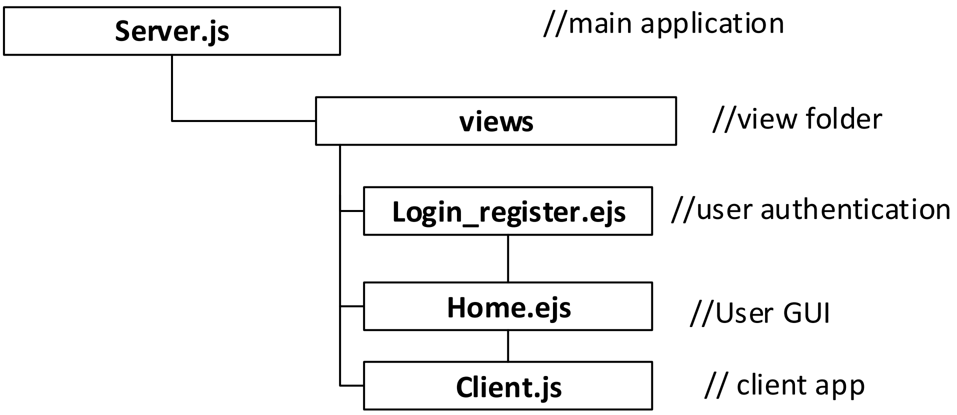

Figure 4.3.1-1 Demo application structure

Login Page simple html code as shown bellow and the entire application structure as follows

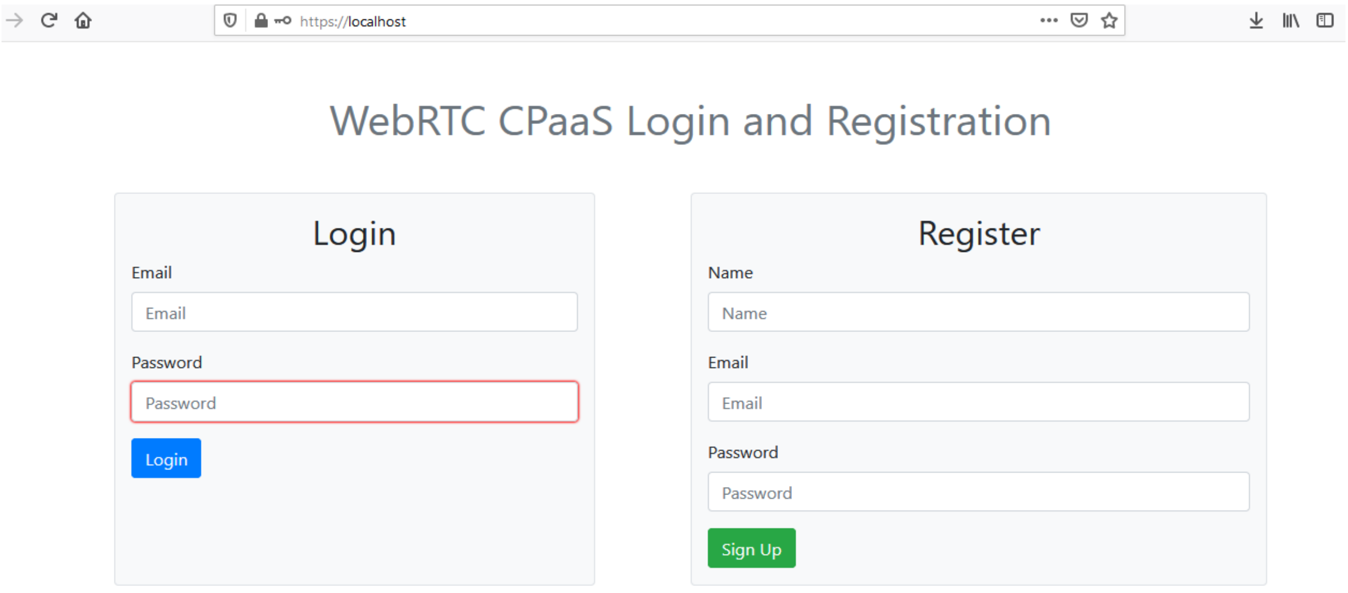

Figure 4.3.1-2 Login registration page

Main Page.

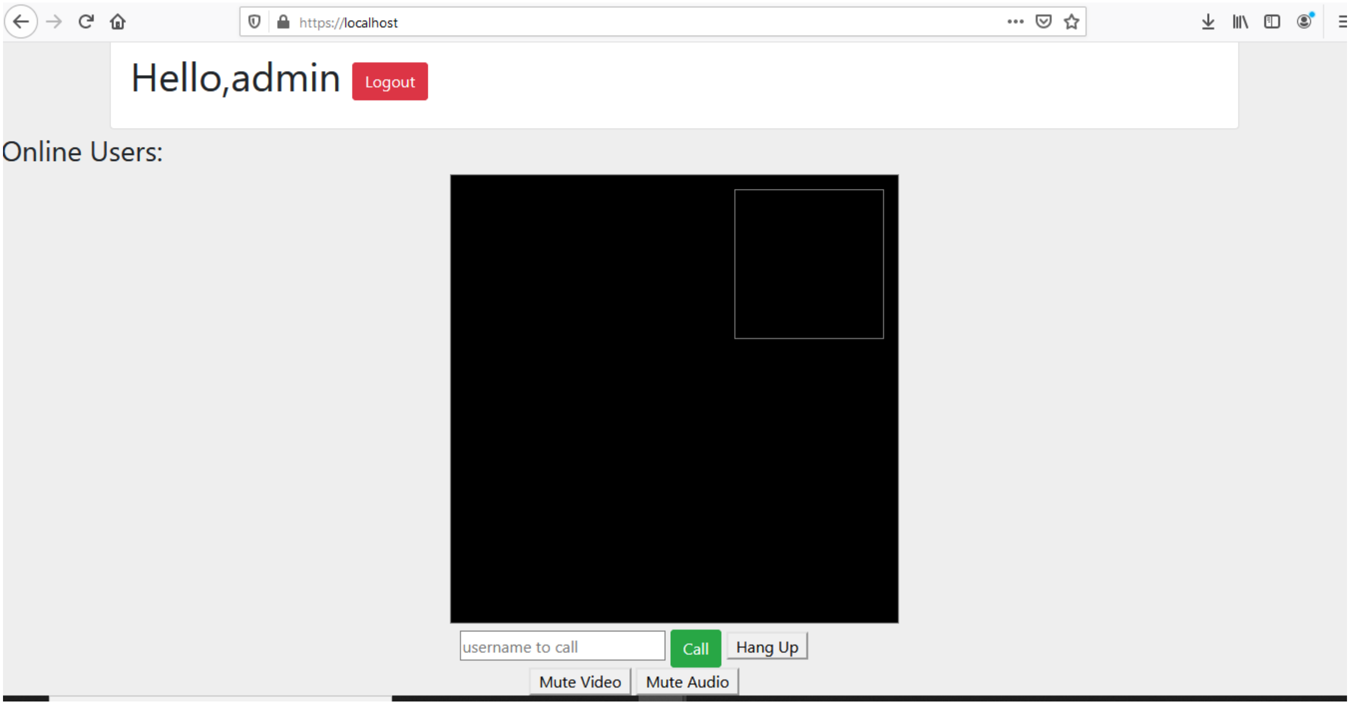

Figure 4.3.1-3 WebRTC application main page

4.3.2 **WebRTC API code**

For signaling ICE STUN and TURN server we used Coturn which is a free and open-source implementation of a TURN and STUN server for VoIP and WebRTC developed for linux based system.

WebRTC allow a real time peer to peer communication between two devices,however, communication is established by a discovery and negotiation process called signaling. The signaling server will use a secure communication https for web and wss for websocket connection, for this test we used a self signed certificate. First the Server.js will add Nodejs liberaries

- Express: is a minimum web application framework for Node.js provides many set of features.
- WebSokect:Websocket library used for bidirectional communication as a signaling rendezvous.
- HTTPS: is the HTTP protocol over TLS/SSL
- FS: File I/O

In this impletation the web application will be running on https tls/ssl protocol express application whick makes a websocket server runs on https as a secure websocket(wss).

Thus the server side code prepared in javascript that will be implemented on the runtime environment Nodejs listed as below:-

```
const express = require('express');
const WebSocket = require('ws');
const WebSocketServer = WebSocket.Server;
const https = require('https');
const fs = require('fs');
const port = 443;
var key = fs.readFileSync(__dirname + '/certs/key.pem');
var cert = fs.readFileSync(__dirname + '/certs/cert.pem');
var options = {  key: key,  cert: cert };

const app = express();
app.use(express.urlencoded({extended:false}));

const https_server = https.createServer(options, app);
https_server.listen(port,'0.0.0.0');
const wss = new WebSocketServer({server: https_server});


wss.on('connection', function(ws) {
    ws.on('message', function(message) {
 // Broadcast any received message to all clients
 console.log('received: %s', message);
wss.broadcast(message);
   });
});

wss.broadcast = function(data) {
this.clients.forEach(function(client) {
  if(client.readyState === WebSocket.OPEN) {
    client.send(data);     }
});
};
```

4.4 Experiment setup

Network set up will be to simulate the service provider network topology in which customer wide area connectivity with there ISP is configured in a point to point manner. The test cases main goal is to to simulate real world internet latency and bandwith constraints on WebRTC communication.

4.4.1 Dummynet configuration and WebRTC analyzer

Dummynet[45][46] is used to configure network limitation, dummynet is a live network emulation tool, originally designed for simulating internet communication, and it can be used for a different type of application programs also as a bandwidth management. It simulates/enforces queue and bandwidth limitations, delays, packet losses, and multipath effects. It also implements various scheduling algorithms. dummynet can be configured on the device running the user's program, or on external machines behaving like routers or bridges.[47]

4.4.1.1 Parameter initilizaiton

We can limit available bandwidth to 2 Mbps for both upstream and down stream

1. Bandwith constraint
   - ipfw pipe 1 config bw 2000 Kbit /s
   - ipfw pipe 2 config bw 2000 Kbit /s
2. Video resolution
   - 640x480
3. Browser
   - Chrome
   - Firefox

Browsers allow saving of realtime WebRTC call log dump file in which we use online anlysis tool in graphical and detail call duration analysis[48]. WebRTC dump file will present various call setatistics between end points such as video, audio and ICE changes. Due to its nature, analyzeRTC can be viewed as a feature used by testRTC's testingRTC and up RTC products and as standalone service of its own.

4.4.2 Case I:WebRTC connection with external STUN

In this first experiment we will check our demo WebRTC Application using external free STUN server to observe ISP distance, bandwidth and latency. The external stun server used is google stun server which is a freely available stun service. The goal of the test is to check the response from stun server for the WebRTC client to communicate that lies behind different NAT router such that the routers will find their public ip and port to pass SDP communication. External ICE stun server configured for WebRTC peer connection setup config as an ice servers attribute {'urls': 'stun:stun.l.google.com:19302'}. These google stun server will using port 19302, when WebRTC client request for ICE candiated address the STUN server will relay with a packet using port number and public or gateway IP address behind NAT server.

iceServers.stun	stun.l.google.com:19302
iceTransportPolicy	all
bundlePolicy	balanced
rtcpMuxPolicy	require
iceCandidatePoolSize	0
sdpSemantics	'unified-plan'

Table 4.4.2-1 External STUN server configuration

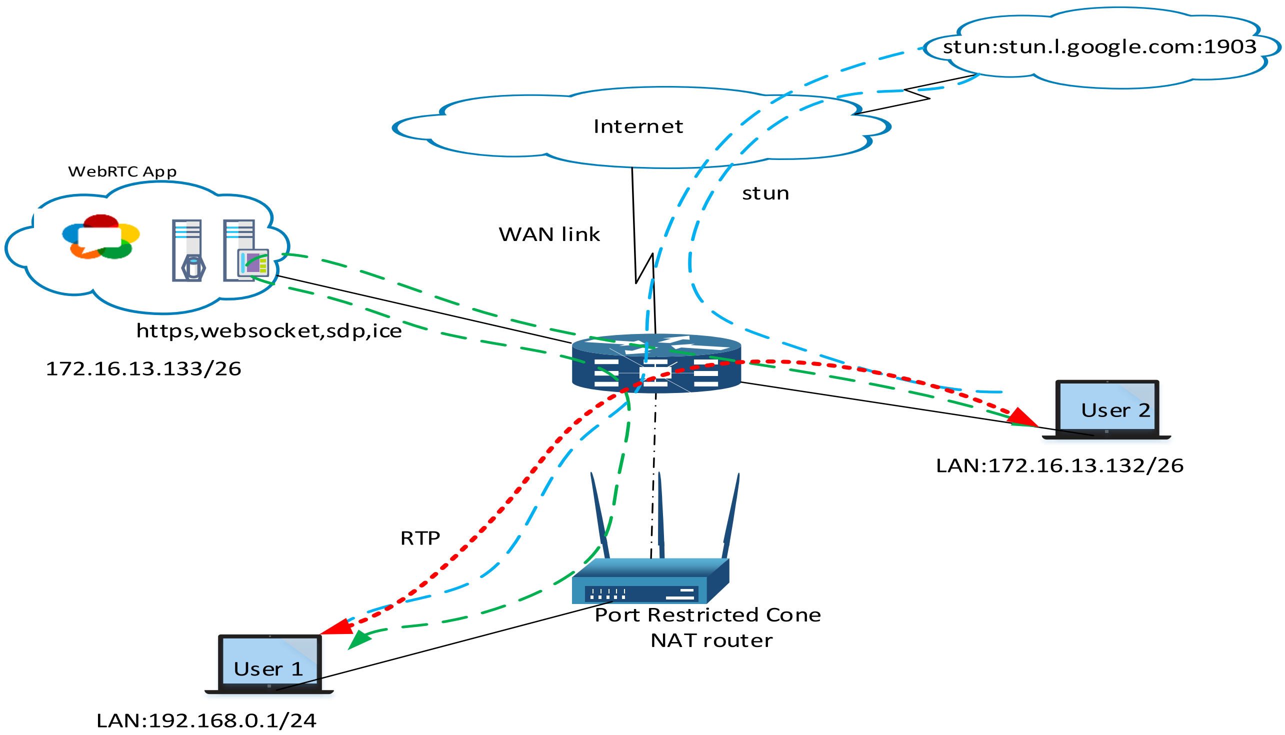

Figure 4.4.2-1 WebRTC App test using external stun server

This WebRTC call session couldn't setup a successful connection because the selected Ice candidate is the same public IP but different port number. NAT server doesn't support port-mapping when accessed from local network. This is one of the case in which currently available WebRTC connection jamps to TURN server alternative. Since we only want to see external STUN server this test case call session will be drop after the JSEP tried for alternate stun or TURN server ICE candidates.

4.4.3 Case II:WebRTC connection setup with local STUN

The throughput tested from a PC1 Google Chrome Web browser to PC2 Mobile Firefox web browser using the different network segment with internal ICE configuration.

iceServers.stun	172.16.13.135:3478
iceTransportPolicy	all
bundlePolicy	balanced
rtcpMuxPolicy	require
iceCandidatePoolSize	0
sdpSemantics	'unified-plan'

Table 4.4.3-1 Local ICE server configuration

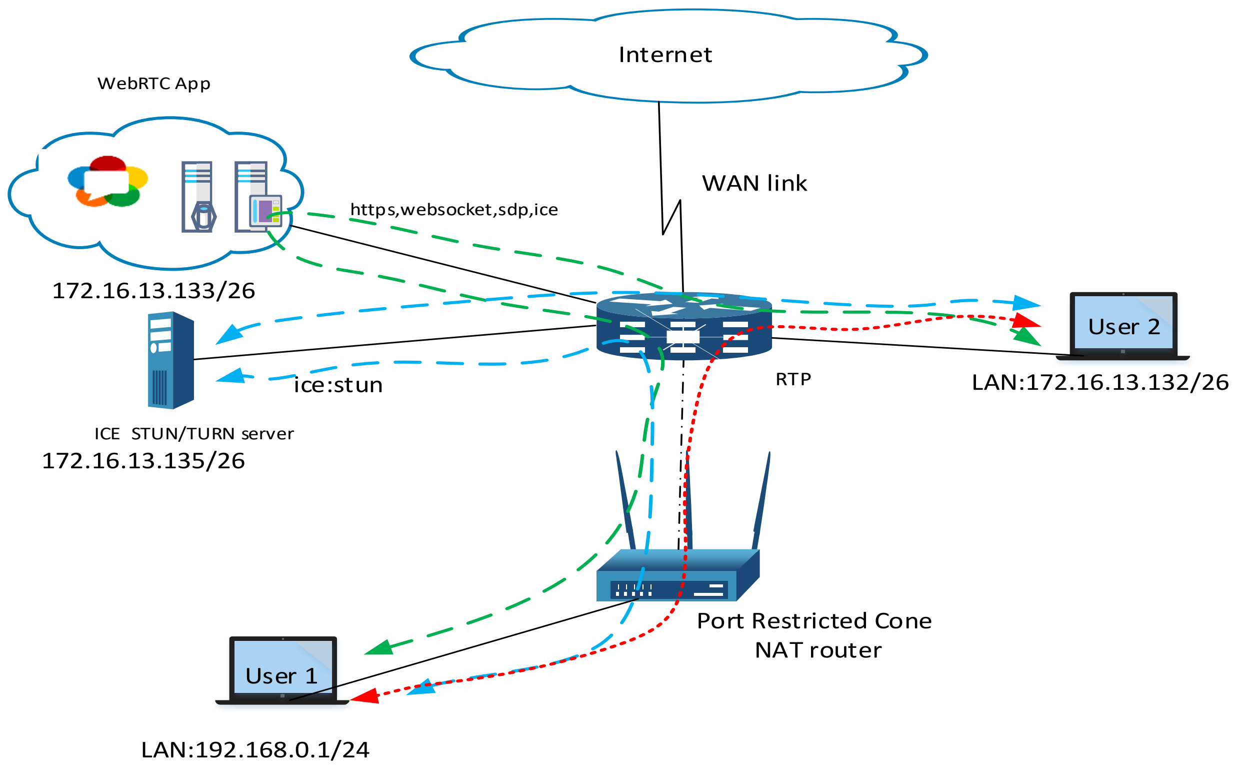

Figure 4.4.3-1 Experiment 2 Topology setup

Using WebRTC’s built in RTCStatsReport to gather performance statistics of all transmited and received data we can find detail information in browsers dump file. The call statistics are gathered at 1 second intervals, in which captured at each end point. Test results are shown below graphically and discussed in the later chapter.

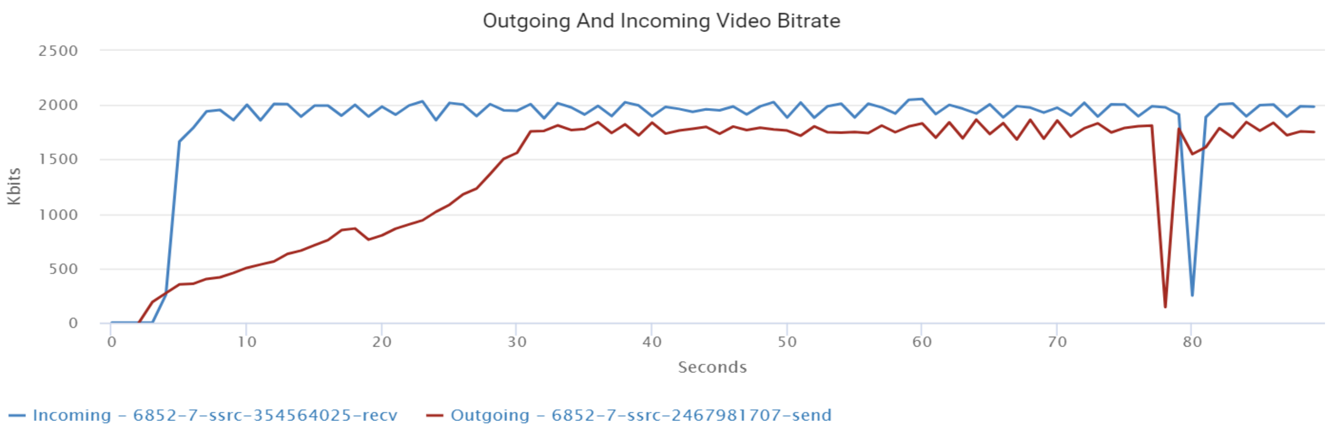

Figure 4.4.3-2 WebRTC Video Bitrate call between user1 and user2 using local STUN ICE

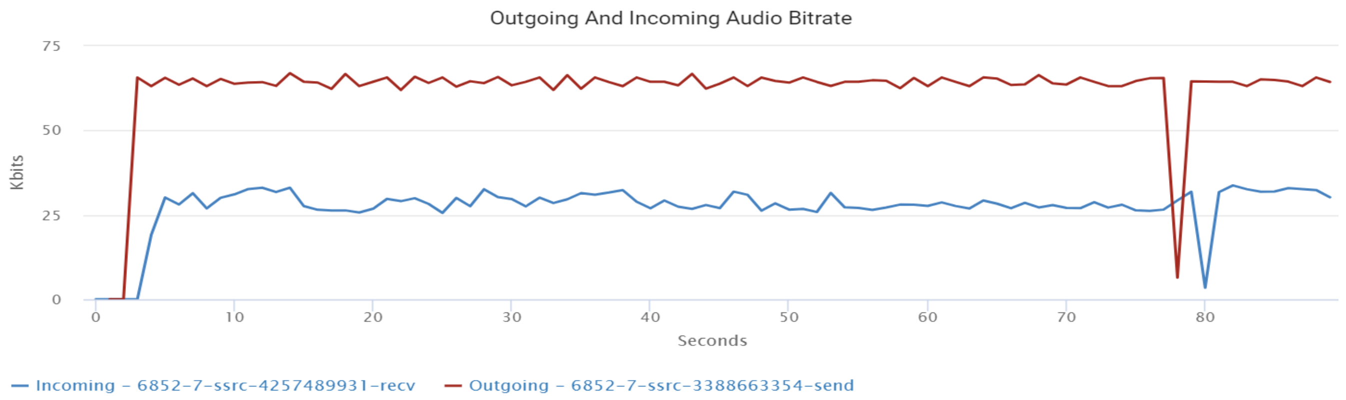

Figure 4.4.3-3 RTC outbound RTP Audio Bitrate

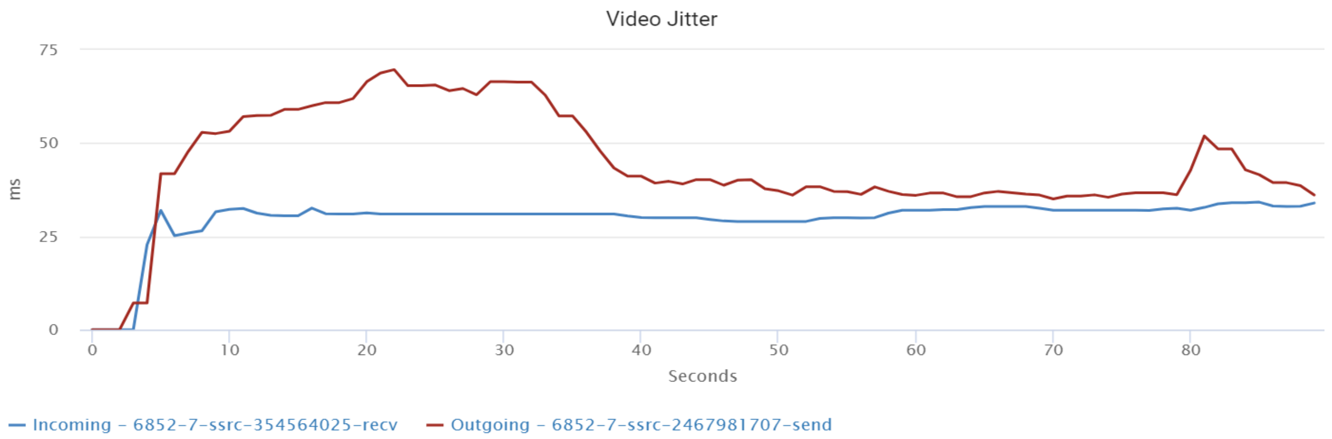

Figure 4.4.3-4 Video jitter for incoming and outgoing traffic with local STUN server

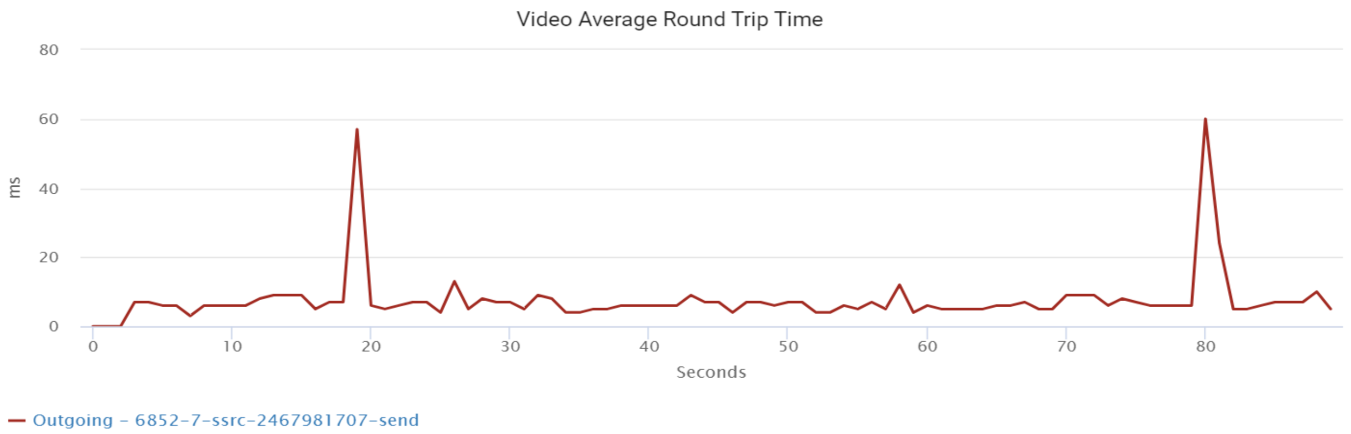

Figure 4.4.3-5 Video Average Round Trip Time for Local STUN server

4.4.4 **Case III:WebRTC connection relay with local TURN server**

In this test case firefox web browser is used for it's unique ability which can force WebRTC traffics only to come via a relay server thus, we will test our TURN ICE server performance. First, using firexfox option “about:config” in address bar as shown in the figure, we will set a boolian attribute of “media.peerconnection.ice.rely_only” setting to true. Therefore, this scenario inspite of the topology forces WebRTC stream to come via TURN server only.

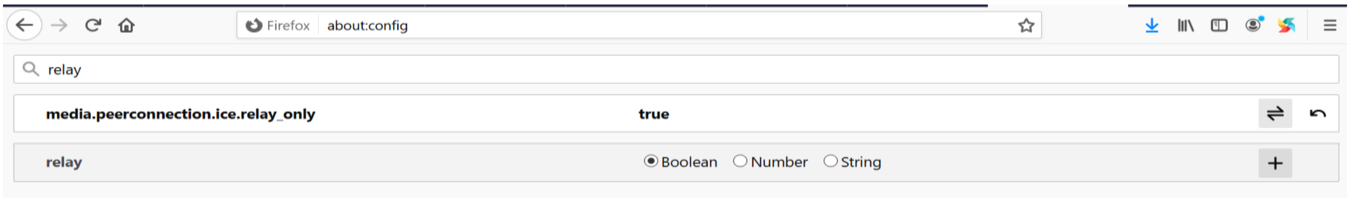

Figure 4.4.4-1 Firefox browser WebRTC relay only configuration.

iceServers.turn	172.16.13.135:3478
iceTransportPolicy	all
bundlePolicy	balanced
rtcpMuxPolicy	require
iceCandidatePoolSize	0
sdpSemantics	'unified-plan'

Table 4.4.4-1 Local TURN server configuration

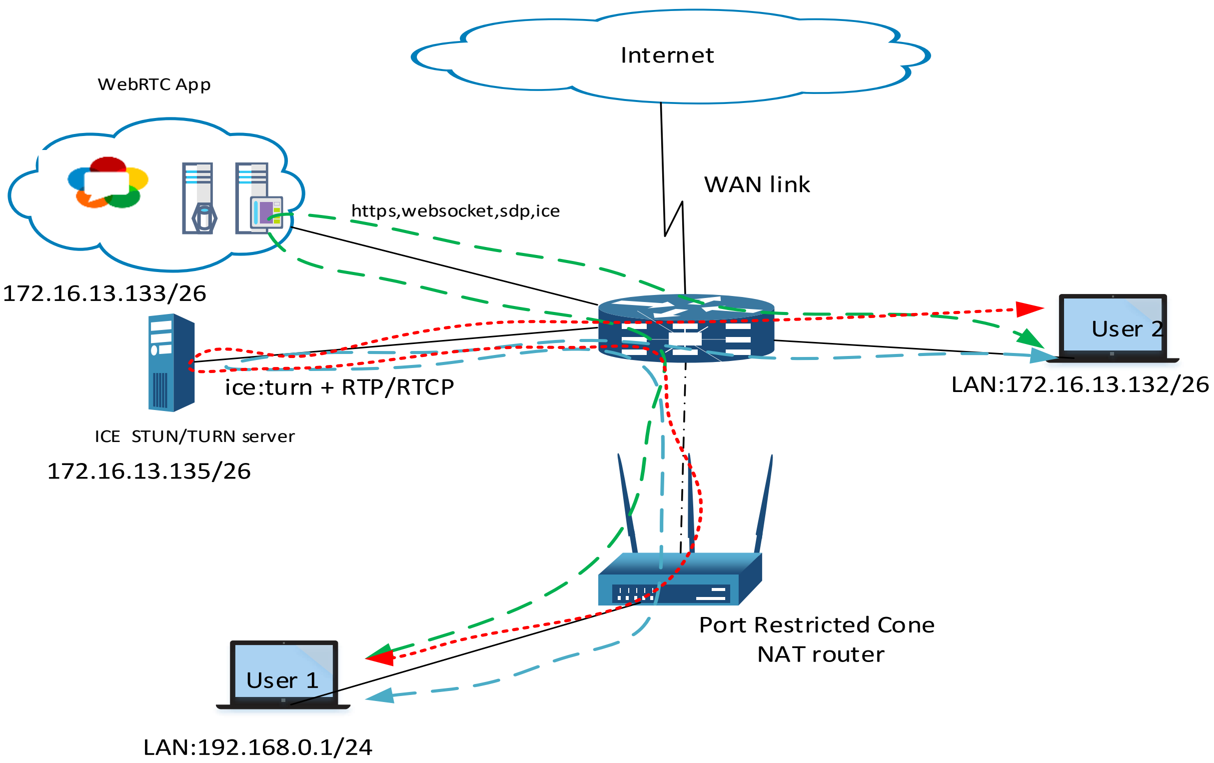

Figure 4.4.4-2 Forced TURN server Relay topology using local TURN server

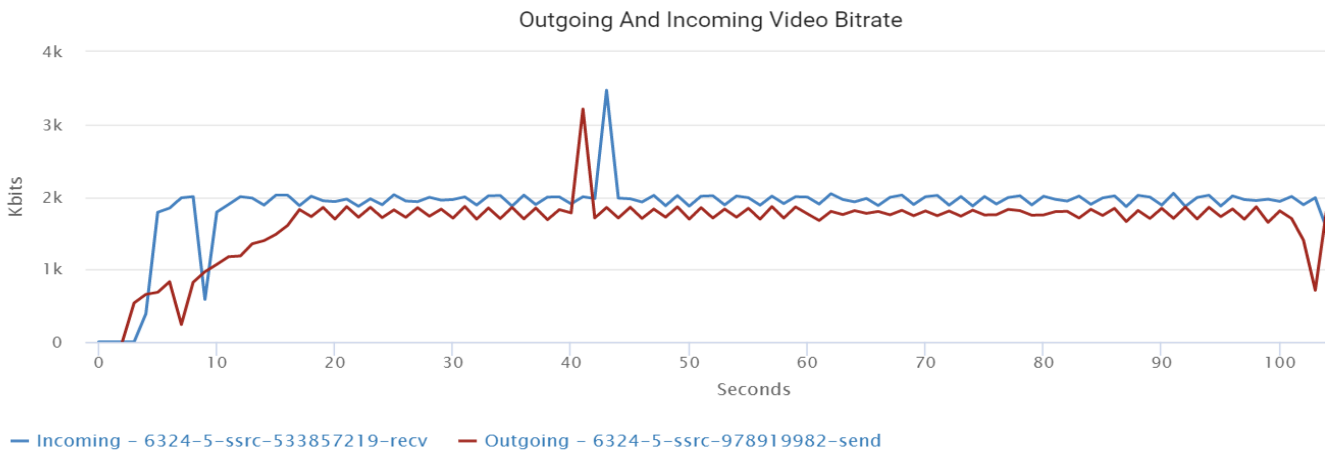

Figure 4.4.4-3 WebRTC call Video Bitrate using forced local TURN ICE

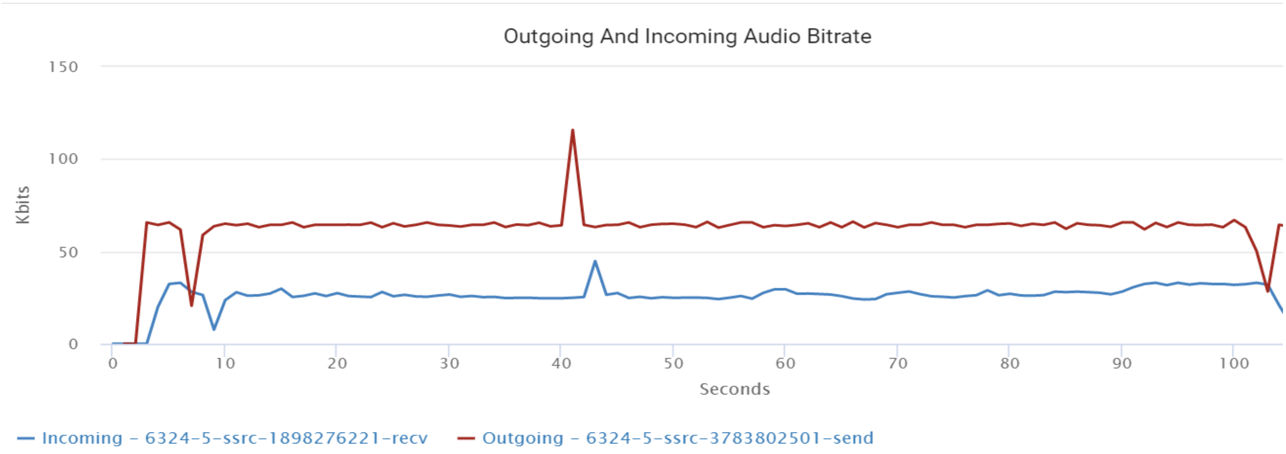

Figure 4.4.4-4 WebRTC incoming and outgoing Audio Bitrate for local TURN server

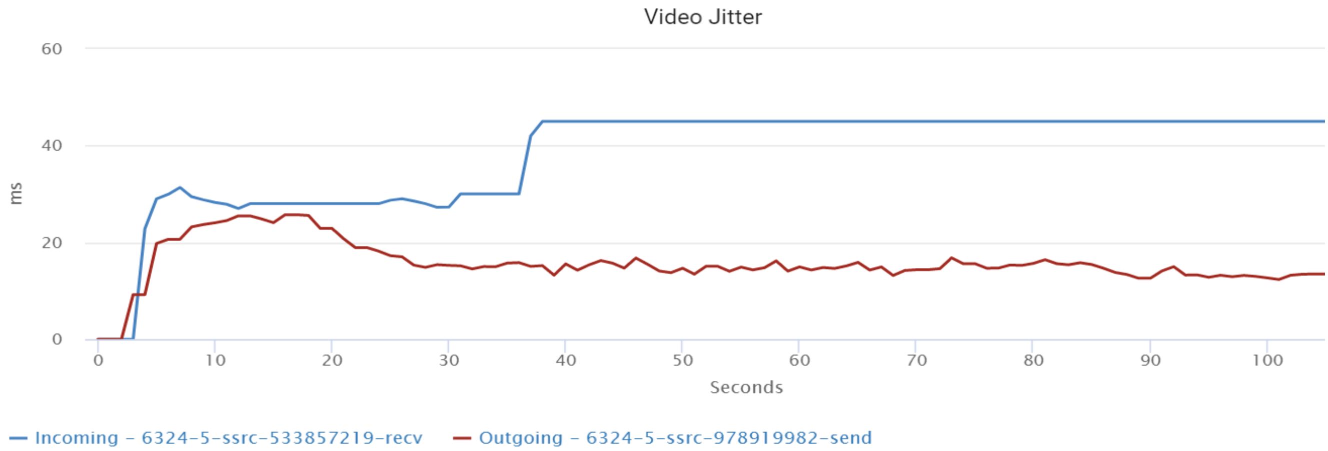

Figure 4.4.4-5 WebRTC jitter for incoming and outgoing traffic using local TURN server

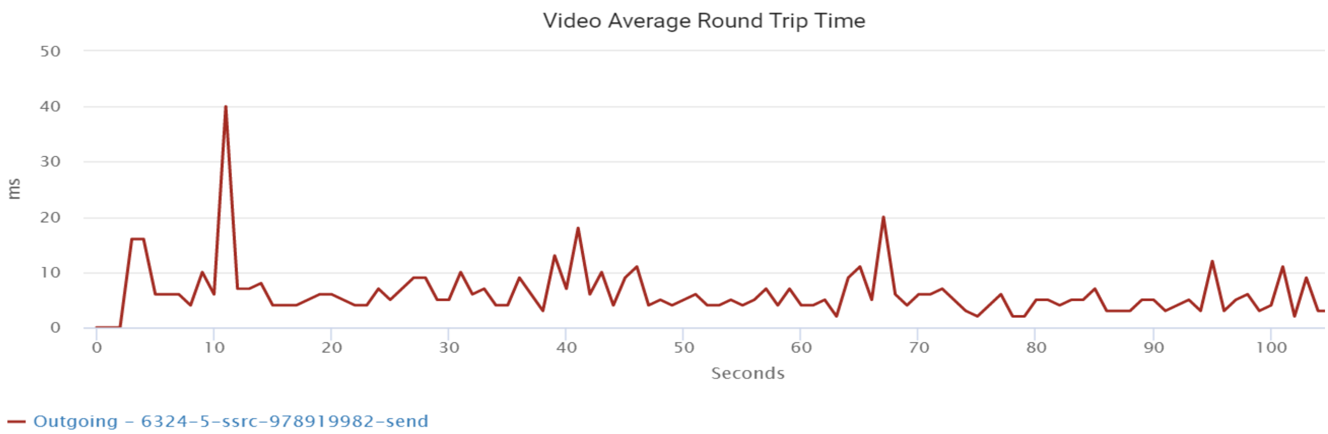

Figure 4.4.4-6 WebRTC average round trip time using local TURN server

4.4.5 Case IV:WebRTC connection relay with external TURN

In this test case we use external free TURN server from Numb [51] from Viagénie which allows to create a free TURN account to use for end-to-end connectivity media relay when STUN server is not an option. Using the above test case III option by forcing firefox browser rtcpeerconnection relay only opotion thist test case tends to measure the performance of external TURN server effect. Viagénie is a consulting and research & development firm with expertise in advanced computer networking technologies, such as Internet Protocol (IPv4 and IPv6) networks and protocols, internationalization, embedded Linux.

iceServers.stun	stun.l.google.com:19302
iceServers.turn	numb.viagenie.ca
iceTransportPolicy	all
bundlePolicy	balanced
rtcpMuxPolicy	require
iceCandidatePoolSize	0
sdpSemantics	'unified-plan'

External ICE TURN server configuration

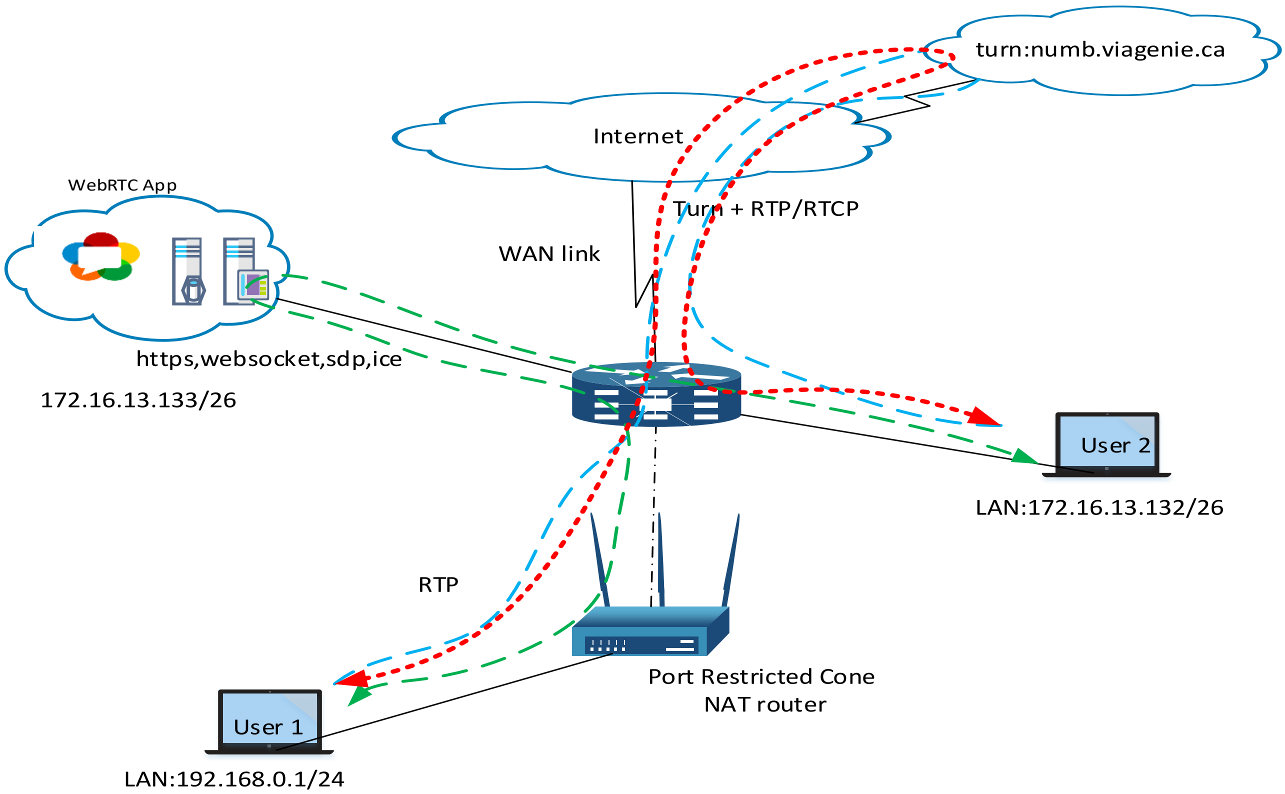

Figure 4.4.5-1 Forced TURN server Relay topology using external TURN server

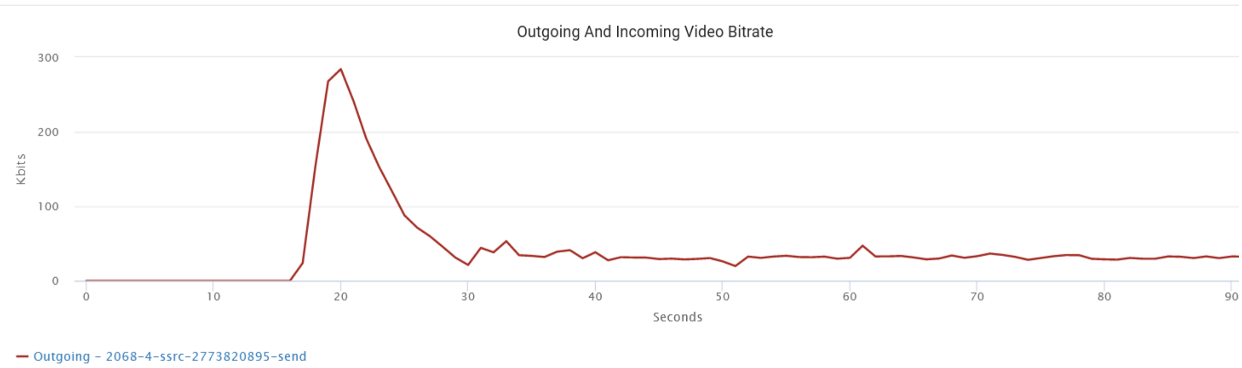

Figure 4.4.5-2 WebRTC Video Bitrate using external TURN ICE

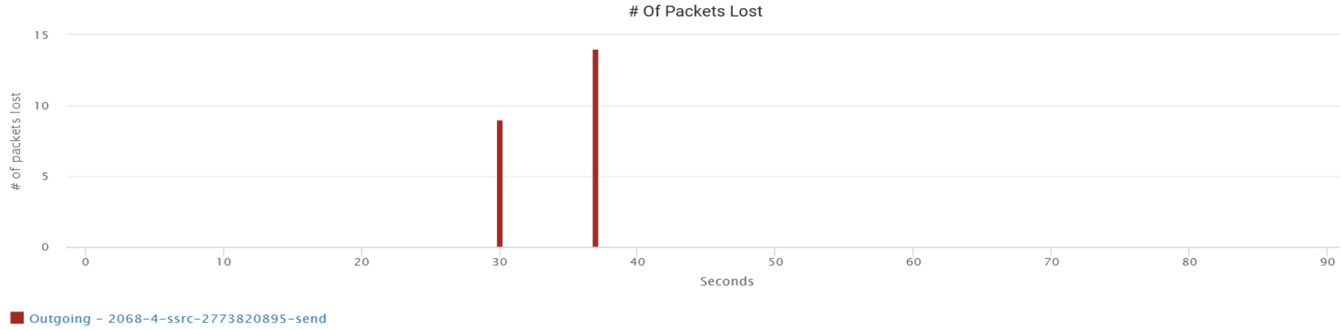

Figure 4.4.5-3WebRTC Packet loss using forced external TURN ICE

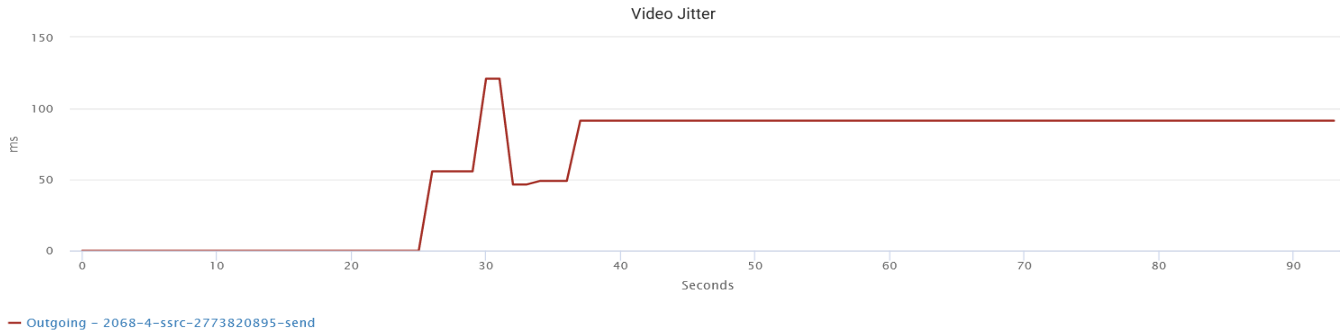

Figure 4.4.5-4 WebRTC Video Jitter using forced external TURN ICE

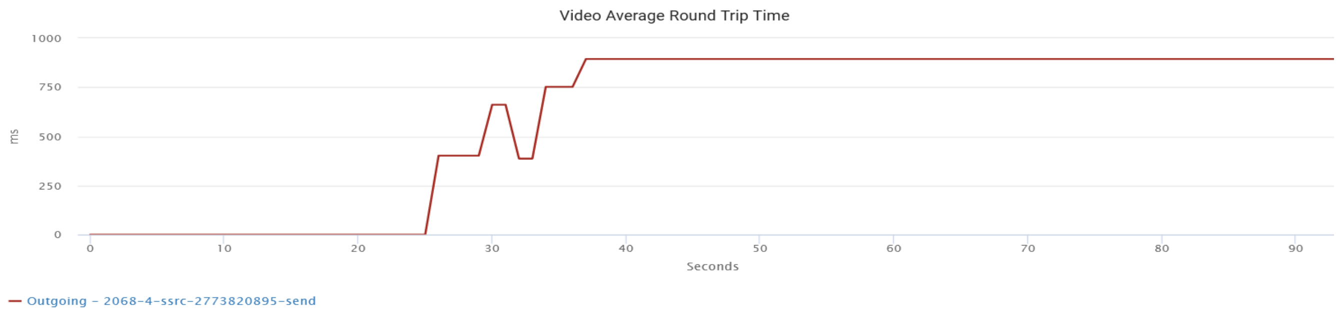

Figure 4.4.5-5 WebRTC Video Average RTT using external TURN ICE

CHAPTER 5

Results and discussion

5.1 Overview

In this chapter we compare the performances of various topology and WebRTC peer to peer communication experiement conducted in chapter four. The results are then discussed in detail and analayze in order to meet the target and objective of the study.

5.2 WebRTC with external STUN server

The fist test case is a problem associated with most currently avialble WebRTC service on the Internet in which the internet service provider network covering a large span of geographical area with one NAT gateway router. This test case clearly shows that beside a network latency and bandwidth problem ICE candidites that doesn’t allow communication due to differnet types of multiple NAT on the path between two client is one of the biggest problem.

In WebRTC call session establishment besides SDP packets exchanging and negotiating media types ICE UDP & TCP packets exchanged between the peers until a working candidate found. ICE UDP candidate types are ‘host,prflx,srflx and relay’, and TCP candidate types are ‘acitve, passive and so’. Each candidate type listed as below:

- ‘host’ – is actual interface ip address for local and remote
- ‘prflx’ – is a peer reflexive is IP address provided from a symmetric NAT
- ‘srflx’ – is a server reflexive IP address provided from a STUN server.
- ‘relay’ – is similar to srflx but provided by TURN server
- ‘active’ – TCP candidate try only outgoing connection and discards incoming connection
- ‘passive’ – TCP candidate accepte incoming connection only and doesn’t start outgoing
- ‘so’ – TCP candidate type try a simultaneous connection

The Experiment case I, we can see that the peers exchange ‘srflx’ candidate type however the NAT type which is a port restricted cone NAT doesn’t allow RTP session so that next ICE relay will be tried.

5.3 WebRTC with local STUN server

The second experiment conducted is to simulate implementation of ICE STUN server inside ISP in which able to reduce the WebRTC signaling latancy significantly as compared to a WebRTC client trying to communicate external OTT based or free servers such as google. During connection setup ice candidate communication STUN server will select “srflx” IP address once successed communication will begin. A total of 17581.648 kb sent packet and 23150.811 kb received data for a total call duration.

Constraints	User 1	User 2	Average	Codec
Video rate (kbps)	1896kpbs	1471kbps	1683.5kpbs	VP8
Audio	28kpbs	62kbps	45kpbs	Opus
RTT (ms)	241 ms	241 ms	241 ms	-
Packet loss(%)	0.00	0.00	0.00	-
Avg received resolution (px)	640x480	640x480	640x480	

Table 4.4.5-1 Output between two devices communication via STUN server.

As it is expected the round trip time and jitter is much better than TURN server, since most of WebRTC communication uses this type of connection scenario, unless symmetric NAT or a combination of two restricted and port restricted cone NAT present along the communication path STUN is the primary choice. Opus audio codic is one of the best available audio codic in which support a low bandwidth voice communications transmission over the Internet developed by Xiph.Org foundation and standardized by the IETF. Opus audio codic developed to efficiently code speech and general audio in a single format. Opus and the full spectrum of what we hear with something like music because it actually combines parts of two type of codecs the first SILK for narrowband and second CELT for wideband. VP8 (Video Compression Format or Video Compression Specification) is an open and royalty-free video compression format developed by On2 Technologies a specification for encoding and decoding high quality video as a file or a stream for display.

In general STUN server implementation inside an ISP network will benefit not only by reducing call time latency rather it will reduce unnecessary NAT translation and Public IP utilization, since each WebRTC client uses Public IP address for communication with out of ISP domain STUN server.

5.4 WebRTC connection relay with local TURN

In this test case a relay server is used such that the jitter is a bit higher than STUN server with the same audio and video codec. Also there is a slight increase in the video rate but audio rate kept constant. In both test case there is no packet loss, but there is relative a high jitter and video round trip. In WebRTC even if rarely used TURN server is the ultimate guaranty for communication if STUN connection fails because of firewall or symmetric NAT.

Constraints	User 1	User 2	Average	Codec
Video rate (kbps)	1901kpbs	1689kbps	1795 kpbs	VP8
Audio	26kpbs	63kbps	44.5 kpbs	Opus
RTT (ms)	332 ms	332 ms	332 ms	-
Packet loss(%)	0.00	0.00	0.00	-
Avg received resolution (px)	640x480	640x480	640x480	

Table 4.4.5-1 Avarage output for a peer-to-peer communication via TURN server.

As the per the proposed solution design TURN server should be located some what a balanced distance from the two WebRTC client in order to avoid latency. If the clients resides outside the ISP, the ISP can rent or implement additional TURN servers anywhere in the Internet such that any client residing in one area should uses TURN sever co-located at that area using Geo-DNS server.

5.5 WebRTC connection relay by external TURN

WebRTC test using external TURN server provided a poor one way connection only with a high packet drop and jitter. This case what most defines current situation in the ISP in which most of the WebRTC communication is made through a TURN relay server with a high latency and jitter. Due to this WebRTC RTP media packets travels unnecessary distances. Using windows 'tracert' command tracing route to the numb.viagenie.ca TURN server shows that the packet travels more than 20 hops or routers with incremental delays. "*C:\Users\user>tracert numb.viagenie.ca*" Distance to this server as well as round trip time is very large for real-time communication.

It is known that most OTT VoIP service used with slow internet connection have similar problem associated with bandwith and speed, but with WebRTC technology adoption this issue can be easily solved.

CHAPTER 6

Conclusion and Future Work

This part of the thesis presents the summary of conclusion of the study and future work for further works on this thesis research area.

6.1 Conclusion

Most traditional telecom operator or internet service provider (ISP) focuses on improving web surfing and web caching technologies, but transmitting voice packets with its packetization algorithm, codics and device delays across the internet is a different process. As such, VoIP calls require certain internet protocols and enhanced QoS in which the normal ISP may not be providing to its customers.

Presently, ISP-OTT providers are getting tough to compete by their constant and foundational continues improvements that the traditional telecoms can't follow up. Even with domestic user source traffics that generated within and destined to inside ISP shouldn't have leaved the ISP and routed back. OTT's online content/service providing business idea have challenged the existing telecoms growth and revenue, requiring operators to consider communications services outside the existing fixed broad band and mobile data device context.

The primary purpose of CPaaS is to be able to add real time communication features to the individual or enterprise existing business application solution. Added features such as voice, video, and SMS can be deployed through communication APIs (Application Program Interfaces) also can be integrated with other communication technology. Companies usually for business that doesn't require an internet such as only VPN (Virtual Private Network) for interconnecting branch offices and their headquarters, have greatly suffered from limitation of communication capabilities. Due to these needs they spent extra unnecessary investment for real time communication software or by adding additional internet connection. As we have seen from this research paper for companies or user who needs online real time communication has to depend on external OTT providers. Thus, the quality of service offered by OTT providers highly affects the users and their internet service provider relation.

Therefore, CPaaS solution not only provides greater benefits by supporting real time communications but provides additional benefits for the service provider by decreasing OTT frauds and isolation of real-time communication traffics on the provider network.

6.2 Contributions

The the thesis paper provides a better understanding concepts in case of real time communication importance technological advaces, features and implementation as a business. WebRTC is a new technology and with existing cloud telechnolgy platform as a service it will provide a huge opportunity and advantage for telecom service providers since they own the infrastructure. ISP's can impact on cloud service in terms of service delivery, QoS and QoE.

The paper also provide cloud provisioning algorithm for CPaaS as an ISP to provide communication platform alternatives for it's customers and as a business opportunity to work with potential OTT and social media. This trend will allow the ISP provider to reduce traffics and cost associated with alternative platforms and OTT fraud related problems.

6.3 Recommendations for Future work

These paper recommends in fact any telecom operators or internet service providers to invest in WebRTC cloud for the advantages it provides. WebRTC is open-source with device and platform independence currently there more than one billion devices with WebRTC capability in which provides business opportunity and alternatives for legacy communications.

WebRTC provides a secure voice and video i.e. has encryption on voice and video DTLS/TLS for signaling and secure RTP protocol (SRTP) for encryption and authentication of both video and voice. This feature of WebRTC protects communication from eavesdropping and recording of the video and voice typically useful for WiFi connections.

This research help and brings interest in the case of WebRTC cloud communication platforms and the scope of future work of research area is as following: -

- WebRTC signaling improvement and security.
- WebRTC integration and communication with the legacy networks towards unified communication
- Doing traffic analysis based on service types like mobile and fixed broad band
- End-to-end Synchronization and packet delay analysis.
- Integration with multiple tenants and external ISP CPaaS clouds.

References

[1] "WebRTC 1.0: Real-time Communication Between Browsers". World Wide Web Consortium. 27 September 2018. Retrieved 08 April 2020. https://www.w3.org/TR/webrtc/

[2] John Naughton (2016) The evolution of the Internet: from military experiment to General Purpose Technology, Journal of Cyber Policy, 1:1, 5-28, DOI:10.1080/23738871.2016.1157619

[3] Joshi Sujata, Sarkar Sohag, Dewan Tanu, Dharmani Chintan, Purohit Shubham and Gandhi Sumi, Impact of Over the Top (OTT) Services on Telecom Service Providers Indian Journal of Science and Technology, Vol 8(S4), 145–160, February 2015

[4] Worldwide mobile data pricing league (no date) Cable.co.uk. [Internet] Available at: https://www.cable.co.uk/mobiles/worldwide-data-pricing/ (Accessed: October 4, 2020).

[5] ICT Sector in Ethiopia http://www.mcit.gov.et/web/guest/ict-sector-in-ethiopia,14.06.2019

[6] Vijay K. Vaishnavi, William Kuechler, Jr. "Design Science Research Methods and Patterns: Innovating Information and Communication Technology" Second Edition 2015

[7] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," Journal of management information systems, vol. 24, pp. 45-77, 2007.

[8] https://en.wikipedia.org/wiki/Design_science(methodology) Design science (methodology) January 20, 2020

[9] Kimmy, "A Comparative Study of Clouds in cloud computing", International Journal of computer Science and Engineering Technology (IJCSET), ISSN (Online): 2229-3345, Volume 4, 6 June 2013.

[10] Kirmizioglu, R.A. and Tekalp, A.M., 2019, November. Multi-party WebRTC as a Managed Service over Multi-Operator SDN. In International Conference on Advanced Communication Systems and Information Security (pp. 25-37). Springer, Cham.

[11] Levent-Levi, T. (2016) How different WebRTC multiparty video conferencing technologies look like on the wire, Testrtc.com. Available at: https://testrtc.com/different-multiparty-video-conferencing/ (Accessed: December 10, 2020).

[12] A. Keranen, C. Holmberg, J. Rosenberg, “Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal”, Internet Engineering Task Force (IETF) ISSN: 2070-1721 July 2018.

[13] I. Fette, A. Melnikov, “The WebSocket Protocol” , Internet Engineering Task Force (IETF)ISSN: 2070-1721 December 2011

[14] SIP, I., 2002. Session Initiation Protocol. Internet Engineering Task Force (IETF).

[15] Uberti, J., Jennings, C. and Rescorla, E., 2012. Javascript session establishment protocol. draft-ietf-rtcweb-jsep-01.

[16] Tsirtsis, G. and Srisuresh, P., 2000. RFC2766: Network Address Translation-Protocol Translation (NAT-PT). Feb, 2000.

[17] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson,” RTP: A Transport Protocol for Real-Time Applications”, RFC 3350, July 2003

[18] Zurawski, Richard, "RTP, RTCP and RTSP protocols". The industrial information technology handbook. CRC Press. pp. 28–7. ISBN 978-0-8493-1985-3. (2004).

[19] Albert Abelló Lozano,” Performance analysis of topologies for Web-based Real-Time Communication (WebRTC)”, Master’s Thesis, School of Electrical Engineering, Aalto university 15.8.2013.

[20] Jean-Charles Grégoire,” SDP-based Signaling and OTT Services: History and Perspective on an Evolving Trend”, 19th International ICIN Conference - Innovations in Clouds, Internet and Networks - March 1-3, 2016, Paris.

[21] File:Internet Connectivity Distribution & Core.svg (no date) Wikimedia.org. Available at: https://commons.wikimedia.org/wiki/File:Internet_Connectivity_Distribution_%26_Core.svg (Accessed: October 31, 2020).

[22] Goyal, Sumit. "Public vs private vs hybrid vs community-cloud computing: a critical review." International Journal of Computer Network and Information Security 6.3 (2014): 20.

[23] Goyal, S., 2014. Public vs private vs hybrid vs community-cloud computing: a critical review. International Journal of Computer Network and Information Security, 6(3), p.20.

[24] Boubendir, A., Bertin, E. and Simoni, N., 2016, April. On-demand dynamic network service deployment over NaaS architecture. In NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium (pp. 1023-1024). IEEE.

[25] Garcia, B., López, L., Gortázar, F., Gallego, M. and Carella, G.A., 2017, October. NUBOMEDIA: The first open source WebRTC PaaS. In Proceedings of the 25th ACM international conference on Multimedia (pp. 1205-1208).

[26] Xhagjika, V., Escoda, O.D., Navarro, L. and Vlassov, V., 2017, May. Load and video performance patterns of a cloud based webrtc architecture. In 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID) (pp. 739-744). IEEE.

[27] Janczukowicz, E., Braud, A., Tuffin, S., Fromentoux, G., Bouabdallah, A. and Bonnin, J.M., 2015, April. Specialized network services for WebRTC: TURN-based architecture proposal. In Proceedings of the 1st Workshop on All-Web Real-Time Systems (pp. 1-6).

[28] Telchemy, I., 2006. Impact of Delay in Voice over IP Services. Telchemy Application Notes, Series VoIP Performance Management, 1(7).

[29] Prof. Ramesh Sharda, Prof. Dr. Stefan VoB, "Telecommunications Planning: Innovations in pricing, network" Design and management operations research/computer science interfaces series, Springer 2006

[30] Prof. Ted G. Lewis, "NETWORK SCIENCE: Theory and Practice", Willy, 2009

[31] Tewodros Hailu," Network Traffic Classification Using Machine Learning: A Step Towards Over-the-Top Bypass Fraud Detection", School of Electrical and Computer Engineering, Addis Ababa Institute of Technology, November 14, 2018.

[32] Xiao Chen," Unified Communication and WebRTC", Master's Thesis Norwegian University of Science and Technology, Department of Telematics, Communication Networks and Networked Services June 2014.

[33] M. Sahin and A. Francillon, "Over-the-Top bypass: Study of a recent telephony fraud," in Proc. SIGSAC Conf. on Comp. and Comm. Security, ACM, 2016, pp. 1106–1117.

[34] Dan C. Marinescu," Cloud Computing: Theory and Practice" Second Edition, Elsevier 2018

[35] Deep Medhi, Karthik Ramasamy, "Network Routing Algorithms, Protocols, and Architectures" Second Edition, Elsevier 2018.

[36]Norton, W.B., 2001, May. Internet service providers and peering. In Proceedings of NANOG (Vol. 19, pp. 1-17).

[37] Rec, I.T.U., 2003. Y. 1541: Network performance objectives for ip-based services. International Telecommunication Union, ITU-T.

[38] Abbas, S., Mosbah, M. and Zemmari, A., 1996. ITU-T Recommendation G. 114,“One way transmission time. In In International Conference on Dynamics in Logistics 2007 (LDIC 2007), Lect. Notes in Comp. Sciences.

[36] PINIKAS, N., 2016. A Webrtc Based Platform for Synchronous Online Collaboration and Screen Casting. Technological Educational Institute of Crete.

[39] Nguyen, D.T., Nguyen, K.K., Khazri, S. and Cheriet, M., 2016, September. Real-time optimized nfv architecture for internetworking webrtc and ims. In 2016 17th International Telecommunications Network Strategy and Planning Symposium (Networks) (pp. 81-88). IEEE.

[40] Nuño, P., Bulnes, F.G., Granda, J.C., Suárez, F.J. and García, D.F., 2018, July. A scalable WebRTC platform based on open technologies. In 2018 International Conference on Computer, Information and Telecommunication Systems (CITS) (pp. 1-5). IEEE.

[41] Boubendir, A., Bertin, E. and Simoni, N., 2015, September. Network as-a-service: The WebRTC case: How SDN & NFV set a solid Telco-OTT groundwork. In 2015 6th International Conference on the Network of the Future (NOF) (pp. 1-3). IEEE.

[42] Bertin, E., Cubaud, S., Tuffin, S., Crespi, N. and Beltran, V., 2013, October. WebRTC, the day after: What's next for conversational services? In 2013 17th International Conference on Intelligence in Next Generation Networks (ICIN) (pp. 46-52). IEEE.

[43] Maruschke, M., Haensge, K., Zimmermann, J. and Bach, T., The Role of QoS in WebRTC and IMS-based IPTV Services.

[44] Wikipedia contributors. Design science (methodology). Wikipedia, The Free Encyclopedia. 2020 Nov 11 [accessed 2020 Nov 20].https://en.wikipedia.org/w/index.php?title= Design_science_ (methodology)&oldid=988119382

[45] *Disruptive Telephony* (no date) *Disruptivetelephony.com*. Available at: https://www.disruptivetelephony.com/2012/09/how-webrtc-will-fundamentally-disrupt-telecom-and-change-the-internet.html (Accessed: November 25, 2020).

[46] Rizzo, L., 1997. Dummynet: a simple approach to the evaluation of network protocols. *ACM SIGCOMM Computer Communication Review*, *27*(1), pp.31-41.

[47] Carbone, M. and Rizzo, L., 2010. Dummynet revisited. ACM SIGCOMM Computer Communication Review, 40(2), pp.12-20.

[48] Dummynet. Unipi.it. [accessed 2020 Dec 13]. http://info.iet.unipi.it/~luigi/dummynet/[online]

[49] testRTC. Testrtc.com. [accessed 2020 Dec 13]. https://app.testrtc.com/app/analyzeDump [online]

[50] Numb STUN/TURN Server. Viagenie.ca. [accessed 2021 Dec 21]. https://numb.viagenie.ca/[online]

[51] CloudSim Plus. Cloudsimplus.org. [accessed 2021 Dec 23]. https://cloudsimplus.org/

APPENDIX

6.4 Appendix

6.4.1 Appendix A: WebRTC Signaling STUN & TURN setup

Prerequsit Linux OS and Network to Internet

First update the system and once complete install coturn ICE stun & turn server and nodejs runtime for javascript.

- ➔ Sudo apt-get update
- ➔ Sudo apt-get install coturn –y
- ➔ Sudo apt-get install nodejs -y

Once installed the coturn sever we must stop and do the initial configurations and will start it automatically once configrauation is finished

- ➔ systemctl stop coturn
- ➔ gedit /etc/default/coturn

Uncomment to enable turn server on the line that contains.

TURNSERVER_ENABLED=1

Save change and close for next configuration

Next open coturn configuration file with any text editor and make the following basic changes

- ➔ gedit /etc/turnserver.conf
- ➔ listening-port=<port> // usually 3478 configured

For secured communication

- ➔ tls-listening-port=<port> // usually 5349
- ➔ fingerprint //uncomment
- ➔ lt-cred-mech //uncomment

6.4.2 Appendix B: CloudSim Plus new Data Center Provisioning example code and Output

```
package webrtc;

import org.cloudbus.cloudsim.hosts.Host;
import org.cloudbus.cloudsim.hosts.HostSimple;
import org.cloudbus.cloudsim.resources.Pe;
import org.cloudbus.cloudsim.resources.PeSimple;
import org.cloudbus.cloudsim.utilizationmodels.UtilizationModelDynamic;
import org.cloudbus.cloudsim.vms.Vm;
import org.cloudbus.cloudsim.vms.VmSimple;
import org.cloudsimplus.builders.tables.CloudletsTableBuilder;
import org.cloudbus.cloudsim.brokers.DatacenterBroker;
import org.cloudbus.cloudsim.brokers.DatacenterBrokerSimple;
import org.cloudbus.cloudsim.cloudlets.Cloudlet;
import org.cloudbus.cloudsim.cloudlets.CloudletSimple;
import org.cloudbus.cloudsim.core.CloudSim;
import org.cloudbus.cloudsim.datacenters.Datacenter;
import org.cloudbus.cloudsim.datacenters.DatacenterSimple;

import java.util.ArrayList;
import java.util.List;

public class WebRTC_App {

        private static final int HOSTS = 1;
           private static final int HOST_PES = 8;

           private static final int VMS = 2;
           private static final int VM_PES = 4;

           private static final int CLOUDLETS = 4;
           private static final int CLOUDLET_PES = 2;
           private static final int CLOUDLET_LENGTH = 10000;

           private final CloudSim simulation;
           private DatacenterBroker broker0;
           private List<Vm> vmList;
           private List<Cloudlet> cloudletList;
           private Datacenter datacenter0;



       public static void main(String[] args) {

        System.out.print("Starting......." + "\n" + "..........."+"\n");
        new WebRTC_App ();
       }

       private WebRTC_App () {
              simulation = new CloudSim();
               datacenter0 = createDatacenter();

               //Brocker created for managing vm and cloudlet
               broker0 = new DatacenterBrokerSimple(simulation);
```

```
                vmList = createVms();
                cloudletList = createCloudlets();
                broker0.submitVmList(vmList);
                broker0.submitCloudletList(cloudletList);

                simulation.start();

                final List<Cloudlet> finishedCloudlets =
broker0.getCloudletFinishedList();
                new CloudletsTableBuilder(finishedCloudlets).build();
        }

    private Datacenter createDatacenter() {
        final List<Host> hostList = new ArrayList<Host>(HOSTS);
        for(int i = 0; i < HOSTS; i++) {
            Host host = createHost();
            hostList.add(host);
        }

        return new DatacenterSimple(simulation, hostList);
    }

    private Host createHost() {
        final List<Pe> peList = new ArrayList<Pe>(HOST_PES);

        for (int i = 0; i < HOST_PES; i++) {

            peList.add(new PeSimple(1000));
        }

        final long ram = 2048; //in Megabytes
        final long bw = 10000; //in Megabits/s
        final long storage = 1000000; //in Megabytes


        return new HostSimple(ram, bw, storage, peList);
    }


    private List<Vm> createVms() {
        final List<Vm> list = new ArrayList<Vm>(VMS);
        for (int i = 0; i < VMS; i++) {
            //Uses a CloudletSchedulerTimeShared by default to schedule Cloudlet
            final Vm vm = new VmSimple(1000, VM_PES);
            vm.setRam(512).setBw(1000).setSize(10000);
            list.add(vm);
        }

        return list;
    }


    private List<Cloudlet> createCloudlets() {
        final List<Cloudlet> list = new ArrayList<Cloudlet>(CLOUDLETS);


        final UtilizationModelDynamic utilizationModel = new
UtilizationModelDynamic(0.5);
```

```
        for (int i = 0; i < CLOUDLETS; i++) {
            final Cloudlet cloudlet = new CloudletSimple(CLOUDLET_LENGTH, CLOUDLET_PES,
utilizationModel);
            cloudlet.setSizes(1024);
            list.add(cloudlet);
        }

        return list;
    }

}
```

```
Starting.......
...........
05:51:08,948 |-INFO in ch.qos.logback.classic.LoggerContext[default] - Could NOT find
resource [logback-test.xml]
05:51:08,949 |-INFO in ch.qos.logback.classic.LoggerContext[default] - Could NOT find
resource [logback.groovy]
05:51:08,950 |-INFO in ch.qos.logback.classic.LoggerContext[default] - Found resource
[logback.xml] at [jar:file:/C:/Users/Admin/eclipse-workspace/WebRTC-CPaaS/lib/cloudsim-
plus-6.0.0.jar!/logback.xml]
05:51:08,950 |-WARN in ch.qos.logback.classic.LoggerContext[default] - Resource
[logback.xml] occurs multiple times on the classpath.
05:51:08,950 |-WARN in ch.qos.logback.classic.LoggerContext[default] - Resource
[logback.xml] occurs at
[jar:file:/C:/Users/Admin/.m2/repository/org/cloudsimplus/cloudsim-plus/6.0.0/cloudsim-
plus-6.0.0.jar!/logback.xml]
05:51:08,950 |-WARN in ch.qos.logback.classic.LoggerContext[default] - Resource
[logback.xml] occurs at [jar:file:/C:/Users/Admin/eclipse-workspace/WebRTC-
CPaaS/lib/cloudsim-plus-6.0.0.jar!/logback.xml]
05:51:08,981 |-INFO in ch.qos.logback.core.joran.spi.ConfigurationWatchList@167fdd33 -
URL [jar:file:/C:/Users/Admin/eclipse-workspace/WebRTC-CPaaS/lib/cloudsim-plus-
6.0.0.jar!/logback.xml] is not of type file
05:51:09,105 |-INFO in ch.qos.logback.classic.joran.action.ConfigurationAction - debug
attribute not set
05:51:09,106 |-INFO in ch.qos.logback.core.joran.action.AppenderAction - About to
instantiate appender of type [ch.qos.logback.core.ConsoleAppender]
05:51:09,130 |-INFO in ch.qos.logback.core.joran.action.AppenderAction - Naming appender
as [STDOUT]
05:51:09,160 |-INFO in ch.qos.logback.core.joran.action.NestedComplexPropertyIA -
Assuming default type [ch.qos.logback.classic.encoder.PatternLayoutEncoder] for
[encoder] property
05:51:09,263 |-INFO in ch.qos.logback.classic.joran.action.RootLoggerAction - Setting
level of ROOT logger to DEBUG
05:51:09,264 |-INFO in ch.qos.logback.core.joran.action.AppenderRefAction - Attaching
appender named [STDOUT] to Logger[ROOT]
05:51:09,265 |-INFO in ch.qos.logback.classic.joran.action.ConfigurationAction - End of
configuration.
05:51:09,266 |-INFO in ch.qos.logback.classic.joran.JoranConfigurator@1e965684 -
Registering current configuration as safe fallback point

_[34mINFO
================== Starting CloudSim Plus 6.0.0 ==================
```

```
_[0;39m_[34mINFO  0.00: DatacenterSimple1 is starting...
_[0;39m_[34mINFO  DatacenterBrokerSimple2 is starting...
_[0;39m_[34mINFO  Entities started.
_[0;39m_[34mINFO  0.00: DatacenterBrokerSimple2: List of 1 datacenters(s) received.
_[0;39m_[34mINFO  0.00: DatacenterBrokerSimple2: Trying to create Vm 0 in
DatacenterSimple1
_[0;39m_[34mINFO  0.00: DatacenterBrokerSimple2: Trying to create Vm 1 in
DatacenterSimple1
_[0;39m_[34mINFO  0.00: VmAllocationPolicySimple: Vm 0 has been allocated to Host 0/DC 1
_[0;39m_[34mINFO  0.00: VmAllocationPolicySimple: Vm 1 has been allocated to Host 0/DC 1
_[0;39m_[34mINFO  0.10: DatacenterBrokerSimple2: Sending Cloudlet 0 to Vm 0 in Host 0/DC
1.
_[0;39m_[34mINFO  0.10: DatacenterBrokerSimple2: Sending Cloudlet 1 to Vm 1 in Host 0/DC
1.
_[0;39m_[34mINFO  0.10: DatacenterBrokerSimple2: Sending Cloudlet 2 to Vm 0 in Host 0/DC
1.
_[0;39m_[34mINFO  0.10: DatacenterBrokerSimple2: Sending Cloudlet 3 to Vm 1 in Host 0/DC
1.
_[0;39m_[34mINFO  0.10: DatacenterBrokerSimple2: All waiting Cloudlets submitted to some
VM.
_[0;39m_[34mINFO  20.11: DatacenterBrokerSimple2: Cloudlet 0 finished in Vm 0 and
returned to broker.
_[0;39m_[34mINFO  20.11: DatacenterBrokerSimple2: Cloudlet 2 finished in Vm 0 and
returned to broker.
_[0;39m_[34mINFO  20.11: DatacenterBrokerSimple2: Cloudlet 1 finished in Vm 1 and
returned to broker.
_[0;39m_[34mINFO  20.11: DatacenterBrokerSimple2: Cloudlet 3 finished in Vm 1 and
returned to broker.
_[0;39m_[34mINFO  20.22: Processing last events before simulation shutdown.
_[0;39m_[34mINFO  20.22: DatacenterBrokerSimple2 is shutting down...
_[0;39m_[34mINFO  20.22: DatacenterBrokerSimple2: Requesting Vm 1 destruction.
_[0;39m_[34mINFO  20.22: DatacenterBrokerSimple2: Requesting Vm 0 destruction.
_[0;39m_[34mINFO  20.22: DatacenterSimple: Vm 1 destroyed on Host 0/DC 1.
_[0;39m_[34mINFO  20.22: DatacenterSimple: Vm 0 destroyed on Host 0/DC 1.
_[0;39m_[34mINFO  Simulation: No more future events

_[0;39m_[34mINFO  CloudInformationService0: Notify all CloudSim Plus entities to
shutdown.

_[0;39m_[34mINFO  20.22: DatacenterSimple1 is shutting down...
_[0;39m_[34mINFO
================= Simulation finished at time 20.22 ==================

_[0;39m_[39mDEBUG DeferredQueue >> max size: 3 added to middle: 0 added to tail: 27
_[0;39m

                                         SIMULATION RESULTS

Cloudlet|Status |DC|Host|Host PEs |VM|VM PEs
|CloudletLen|CloudletPEs|StartTime|FinishTime|ExecTime
      ID|       |ID|  ID|CPU cores|ID|CPU cores|        MI|  CPU cores|  Seconds|
Seconds| Seconds
----------------------------------------------------------------------------------------
-------------
       0|SUCCESS| 1|   0|        8| 0|        4|     10000|          2|        0|
20|      20
       2|SUCCESS| 1|   0|        8| 0|        4|     10000|          2|        0|
20|      20
```

```
       1|SUCCESS| 1|   0|         8| 1|         4|      10000|           2|         0|
20|      20
       3|SUCCESS| 1|   0|         8| 1|         4|      10000|           2|         0|
20|      20
-----------------------------------------------------------------------------------------
-------------
```